Oscillation-free Quantization for Low-bit Vision Transformers

Shih-Yang Liu [*1] **Zechun Liu** [*2] **Kwang-Ting Cheng** [1]

Abstract

Weight oscillation is an undesirable side effect of quantization-aware training, in which quantized weights frequently jump between two quantized levels, resulting in training instability and a sub-optimal final model. We discover that the learnable scaling factor, a widely-used *de facto* setting in quantization aggravates weight oscillation. In this study, we investigate the connection between the learnable scaling factor and quantized weight oscillation and use ViT as a case driver to illustrate the findings and remedies. In addition, we also found that the interdependence between quantized weights in *query* and *key* of a self-attention layer makes ViT vulnerable to oscillation. We, therefore, propose three techniques accordingly: statistical weight quantization (StatsQ) to improve quantization robustness compared to the prevalent learnable-scale-based method; confidence-guided annealing (CGA) that freezes the weights with *high confidence* and calms the oscillating weights; and *query-key* reparameterization (QKR) to resolve the query-key intertwined oscillation and mitigate the resulting gradient misestimation. Extensive experiments demonstrate that these proposed techniques successfully abate weight oscillation and consistently achieve substantial accuracy improvement on ImageNet. Specifically, our 2-bit DeiT-T/DeiT-S/Swin-T algorithms outperform the previous state-of-the-art by 9.8%/7.7%/4.64%, respectively. Code and models are available at: `https://github.com/nbasyl/OFQ`.

[*]Equal contribution [1]Hong Kong University of Science and Technology [2]Reality Labs, Meta Inc. Correspondence to: Shih-Yang Liu <sliuau@connect.ust.hk>, Zechun Liu <zechunliu@fb.com>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

Deep neural networks have enjoyed tremendous success in numerous applications, ranging from computer vision (He et al., 2016; Dosovitskiy et al., 2021) to natural language processing (Vaswani et al., 2017; Kenton & Toutanova, 2019). However, the prohibitive model size and resource-intensive computation restrict the feasibility of deploying large models on resource-constrained devices. Among many methods that study the compression and acceleration of neural networks (Liu et al., 2019; Wu et al., 2019; Zhou et al., 2016), quantization-based approaches have stood out due to their high compression ratio and remarkable reduction in throughput time by adopting efficient bitwise operations (Zhu et al., 2020; Zhang et al., 2018a).

Despite the efficacy of quantization, there is still a non-negligible accuracy gap between quantized models and their full-precision counterparts. The accuracy drop comes from several aspects, including but not limited to the discrete nature of quantization and its limited representational capability (Liu et al., 2022; Zhang et al., 2018a; Li et al., 2020; Miyashita et al., 2016), difficulty in gradient approximation to the non-differentiable quantization function (Gong et al., 2019; Liu et al., 2018), and quantization oscillation that hinders optimization (Nagel et al., 2022). The last, relatively under-explored, is the primary focus of this study.

In this study, we choose the quantized ViT as the driver for investigating the cause of oscillation. Previous work (Nagel et al., 2022) assumes the scaling factors are fixed in quantization, while this is not the case as most of the prevalent quantization methods adopt learnable scaling factors, including (Nagel et al., 2022). Surprisingly, we found that the learnable scaling factor exacerbates the oscillation, leading to unstable quantization-aware training and often results in sub-optimal models. The learnable scaling factor is updated with noisy gradients and determines quantization thresholds. Thus, the intertwined oscillation between the noisy learnable threshold and the weights near the threshold makes training unstable.

We visualize the composition of the oscillating weights and observe that they are position-agnostic, meaning that there are no such weights that would persistently stay within the oscillation region during optimization. But the current optimizer fails to prevent new weights from entering the oscil-

lating region while cleaning weights out, so the oscillation cannot subside. Additionally, we find that in a naïvely quantized self-attention layer, oscillation in quantized *query* will directly impact the gradient estimation for the weights in *key* and vice versa, resulting in *query-key* weight co-oscillation.

To this end, we propose three novel techniques based on our analyses to abate quantization oscillation, namely statistical weight quantization (StatsQ) and confidence-guided annealing (CGA) to stabilize training and ultimately eliminate the weight oscillation, and *query-key* reparameterization (QKR) to decouple the negative mutual influence between quantized *query* and *key*.

We demonstrate that the proposed techniques are complementary and capable of working collaboratively to achieve oscillation-free quantized ViTs with consistent improvements over the previous state-of-the-art on ImageNet. Specifically, our quantization method produces 2-bit DeiT-T and DeiT-S that improve the ImageNet top-1 accuracy by 9.88% and 7.72% respectively, compared to previous SoTA, and we show for the first time that 3-bit DeiT-T and DeiT-S can achieve comparable accuracy as the full-precision models.

2. Related Works

Model quantization can be categorized into post-training quantization (PTQ) (Liu et al., 2021c; Nagel et al., 2020; Banner et al., 2019; Nagel et al., 2019) and quantization-aware training (QAT) (Zhou et al., 2016; Choi et al., 2018; Gong et al., 2019). In general, PTQ provides a faster quantization pipeline than QAT, as PTQ relies on calibration data without re-training; on the other hand, QAT is more suitable for precision-sensitive scenarios at the cost of lengthier training time. This work focuses on QAT.

Prior literature has tried to ameliorate QAT from different aspects. Some proposed increasing the quantization representation ability by replacing uniform quantization with non-uniform quantization, such as (Liu et al., 2022; Li et al., 2020; Zhang et al., 2018a). Another line of research suggests reducing the gradient misestimation caused by the non-differentiable rounding function by approximating the discrete quantization with a differentiable function (Gong et al., 2019; Zhou et al., 2016). Moreover, several works have delved into searching for the optimal scaling factor. For instance, (Zhou et al., 2016) incorporates a non-linear function for scaling the weights to restrict their value range. (Choi et al., 2018) introduces a trainable clipping parameter to sort out the suitable quantization scale automatically. Lately, (Esser et al., 2020) introduced a simple, intuitive, yet effective method that resolves around the learnable scaling factor and has been widely adopted as the *de facto* approach for quantization-aware training.

Recently, (Nagel et al., 2022) points out that weight oscillation seriously impacts QAT, mainly rooted in depth-wise convolution (DW-Conv) and the batch normalization (BN) layers. It uses convolution neural networks (CNNs) for the case study. In our study, we found that in vision transformers (ViTs), despite the absence of DW-Conv and BN, the oscillation still exists, which motivates us to choose ViT as a case driver to investigate quantized weight oscillation. In addition, previous work (Nagel et al., 2022) assumed the quantization threshold to be static but neglected the intertwined relation between the learnable scaling factor and weight oscillation. To this end, we delve into studying the entangled relationship between the ViT structure and the oscillation of the quantized weights and understanding the impact of learnable scaling factors on aggravating such oscillation.

3. Preliminary

3.1. Quantization-Aware Training

In quantization-aware training (QAT), the scaling factor is crucial to striking a good trade-off between the representation value range and the quantization step size, especially in low-bit quantization. Previous works (Zhou et al., 2016; Choi et al., 2018; Esser et al., 2020) extensively studied the solution for obtaining a good scaling factor to bridge the accuracy gap between the quantized models and their full precision counterparts. Among these methods, LSQ (Esser et al., 2020) has become the most prevalent method due to its simplicity and effectiveness in learning the scaling factors. Therefore, in this work, we use (Esser et al., 2020) to study the impact of the learnable scaling factor on the oscillation of QAT.

In LSQ (Esser et al., 2020), the quantization function is formulated as:

$$\mathbf{W}_q^{(i)} = \alpha \hat{\mathbf{W}}_q^{(i)} = \alpha \cdot \left\lfloor \mathrm{Clip}\left(\frac{\mathbf{W}^{(i)}}{\alpha}, Q_n, Q_p\right)\right\rceil \quad (1)$$

where $\mathbf{W_q}$ and $\mathbf{W}$ denote the quantized weights and real-valued weights, respectively, and the superscript i denotes the ith entry. α is the learnable scaling factor, and Q_n, Q_p represents the quantization range.

To approximate the gradient through the non-differentiable round function, straight through estimator (STE) (Bengio et al., 2013) is adopted:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(i)}} \overset{STE}{\approx} \frac{\partial \mathcal{L}}{\partial \mathbf{W}_q^{(i)}} \cdot \mathbf{1}_{Q_n \leq \mathbf{W}^{(\mathbf{i})}/\alpha \leq Q_p} \quad (2)$$

where $\mathbf{1}$ represents the indicator function that outputs 1 if $\mathbf{W}^{(\mathbf{i})}/\alpha$ is the inside clipping range and 0 otherwise.

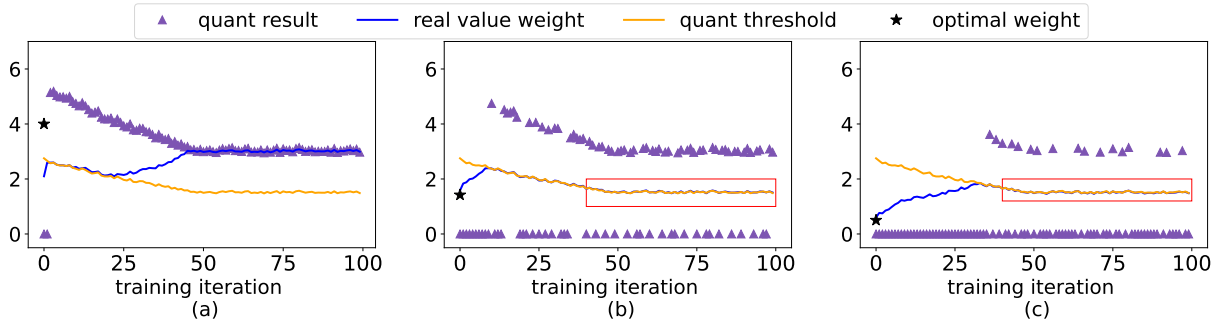

Figure 1: The oscillation of each weight (*i.e.*, (a) $\mathbf{W}^{(1)}$ (b) $\mathbf{W}^{(2)}$, and (c) $\mathbf{W}^{(3)}$) in a 3D toy-regression example. The red frames show that the weights interweave with the learnable scaling factors and exacerbate the oscillation of both parties.

3.2. Quantized ViT Architecture

The main characteristic that differs vision transformer (ViT) from convolutional neural networks (CNNs) is the transformer layer structure, comprising two sub-modules: Multi-head Self-Attention layer (MHSA) and Feed-Forward Network (FFN), which heavily utilize the fully-connected (FC) layers. We adopted row-wise granularity for quantizing weights and activations in FC layers and for quantized activation-activation multiplication in attention layers:

$$\mathbf{X}_q \cdot \mathbf{Y}_q^{\mathrm{T}} = \alpha_{\mathbf{X}_q} \alpha_{\mathbf{Y}_q} \odot (\hat{\mathbf{X}}_q \otimes \hat{\mathbf{Y}}_q^{\mathrm{T}})) \tag{3}$$

where $\mathbf{X}_q$ and $\mathbf{Y}_q$ denote the quantized tensors, $\otimes$ denotes the integer matrix multiplication and $\odot$ denotes the high-precision scalar-tensor multiplication. Note that, the scaling factor should align with the multiplication direction to make Eq. 3 established and facilitate low-cost integer matrix multiplication acceleration (Esser et al., 2020; Xiao et al., 2022). For example, when input $\mathbf{X}_q \in \mathbb{R}^{N \times D_1}$ and $\mathbf{Y}_q \in \mathbb{R}^{D_2 \times D_1}$, we have the scaling factor vector $\alpha_{\mathbf{X}_q} \in \mathbb{R}^N$ and $\alpha_{\mathbf{Y}_q} \in \mathbb{R}^{D_2}$. We denote this as quantizing along the last dimension, where the scaling factor is shared along the last dimension. Specifically, in ViTs, weight and activation matrices are all quantized along the last dimension, except for the *value* matrix $\mathbf{V} \in \mathbb{R}^{N \times D}$ in multiplication with the attention matrix $\mathbf{Attn} \in \mathbb{R}^{N \times N}$, where $\mathbf{V}$ is quantized along the sequence length dimension N and has a scaling factor vector $\alpha_{\mathbf{V}_q} \in \mathbb{R}^D$.

4. Oscillation in QAT

4.1. Oscillation and Learnable Scaling Factor

Despite the prevalence of learnable scaling factor in QAT, its negative impact on training stability is rarely studied. We illustrate the issues by proposing a toy example with three weights $\mathbf{W} = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{W}^{(3)}\}$ and a scaling factor α determining the quantization threshold for all the three weights, following Eq. 1. The optimization objective is a 3D regression problem minimizing the $l2$-loss between the target optimal floating-point weight $\mathbf{W}_* \in \mathbb{R}^3$ and quantized weight $\mathbf{W}_q$ in weighting the data vector $\mathbf{X} \in \mathbb{R}^3$:

$$\min_{\mathbf{W}} \mathcal{L}(\mathbf{W}) = \mathbb{E}_{\mathbf{X} \sim U} \left[\frac{1}{2} (\mathbf{X}\mathbf{W}_* - \mathbf{X}\mathbf{W}_q)^2 \right]. \tag{4}$$

Here $\mathbf{X}$ are sampled from a uniform distribution U on the interval $[0, 1)$. We follow Eq. 2 to optimize the weight.

We observe that the weight oscillation caused by quantization's discrete nature can be greatly amplified by the presence of a learnable scaling factor. In the absence of learnable scaling factors, weights farther from their optimal values tend to be updated more towards the target, while weights close to the optima remain stable. However, we found that this is not the case when learnable scaling factors are present.

As shown in Fig. 1 (b) and (c), weights ($\mathbf{W}^{(2)}$ and $\mathbf{W}^{(3)}$) initialized close to their target values are influenced by the learnable scaling factor and end up oscillating around the quantization threshold. Note that the scaling factor determines the threshold according to Eq. 1. The cause of this phenomenon is the existence of an *outlier* weight ($\mathbf{W}^{(1)}$) that is initialized far from its optima ($\mathbf{W}_*^{(1)}$). The *outlier* weight contributes more to the gradient of the learnable scaling factor, driving it towards an optimal value for this *outlier*, and resulting in the oscillation of the other two weights that are initially set near their optimal values. This observation aligns with the gradient derivation of LSQ (Esser et al., 2020): Weights within the range of $[Q_n, Q_p]$ have limited influence on the scaling factor gradient, with values restricted to between -0.5 and 0.5. On the other hand, weights outside of this range contribute significantly more to the gradient of the scaling factor, e.g. scaled weights that are larger than Q_p contribute Q_p to the gradient of the scaling factor.

More catastrophically, once the weights interweave with the learnable thresholds, as highlighted by the red frame in

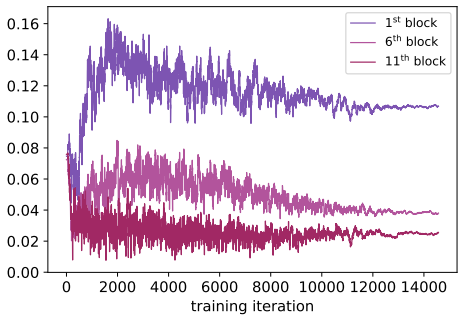

Figure 2: Trajectory of three learnable scaling factors α from the $1^{\text{st}}/6^{\text{th}}/11^{\text{th}}$ transformer blocks in a 2-bit DeiT-T. The y-axis represents the value of α. The fluctuation of learnable scaling factor persists throughout the training.

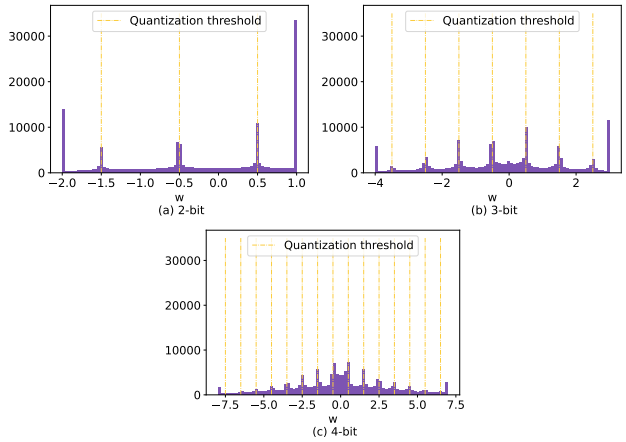

Figure 3: Histogram of the weight distribution in the 1st fully connected layer in the feed-forward network (FFN) in the 8th transformer block of the (a) 2-bit, (b) 3-bit, and (c) 4-bit quantized DeiT-T.

Fig. 1, weight oscillation will introduce noise to the gradient of the learnable scaling factor and cause it to oscillate. The latter, in turn, changes the quantization threshold and aggravates the weight oscillation. This vicious cycle of interaction makes the oscillation hard to subside.

4.2. Oscillations in Quantized ViTs

The aggravation of weight oscillation caused by the learnable scaling factor does not just exist in the toy example. It is a common problem in real-world scenarios. In this section, we choose the vision transformer (ViT) as a case study for quantization oscillations. We analyze weight oscillation in the quantized ViT and examine how the learnable scaling factor exacerbates the oscillation.

4.2.1. The Effect of Learnable Quantization on Quantized ViTs

Throughout the training, we visualize the trajectory of three learnable scaling factors in a quantized DeiT-T (Touvron et al., 2021). Fig. 2 shows that the scaling factors fluctuated drastically, and the fluctuation persists toward the end of training even when the learning rate is small, and the model is supposed to be converged. From our analysis in Sec.4.1, the oscillation of the learnable scaling factor will contribute to weight oscillation. Fig. 3 evidences this by showing clear patterns of a large portion of weights clustered around the quantization threshold. This phenomenon also indicates that there are still non-negligible portions of weights that fail to converge at the end of training and continue to oscillate around the threshold.

We take one step further and inspect the average gradient direction change of weights close to the quantization threshold. We define the term "Boundary Range (BR_x)" as a range that includes all weights $\widetilde{\mathbf{W}}_q$ within the distance x to the nearest quantization threshold where $\widetilde{\mathbf{W}}_q = \mathbf{W}/\alpha$ is the rescaled $\mathbf{W}$ before rounding. Fig. 4 clearly shows that the closer the weight entry is to the quantization threshold, the more frequently its gradient direction flips, and such a phenomenon occurs consistently across all modules in ViT. This phenomenon validates that during oscillation, the gradients of weights near the threshold constantly change directions, push the underlying real value weights to cross the threshold, make the quantized weights jump between two quantized values, and prevent the network from well-converging.

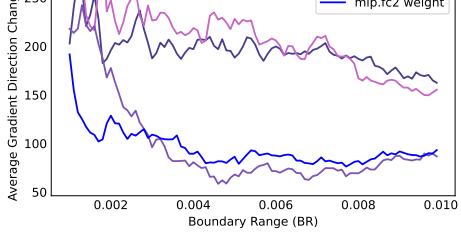

Figure 4: The average number of gradient direction change in 2000 iterations in 2-bit DeiT-T towards the end of training. The frequency of gradient change is anti-correlated with the distance of weights to the quantization thresholds (BR).

4.2.2. Noise Injection Analysis

To further verify our hypothesis that the final quantized network lands at a sub-optima due to weight oscillation within a certain boundary range close to the thresholds, we perform a series of noise injection analyses on quantized

Table 1: Noise injection analysis on quantized DeiT-T. The 1st, 6th, and 11th row are the accuracies of the converged model before the noise injection. "Random" refers to random position noise injection. "within BR" refers to injecting noise only to weights within the boundary. "% of weight" refers to the fraction of the weights with noise injected. μ and σ denote the mean and variance over ten experiment trials.

Method	Val Top-1	% of weights
W2A2	54.20%	–
W2A2 Random ($\mu + \sigma$)	51.85%±0.0873	12.15%
W2A2 Random (best)	52%	12.15%
W2A2 (within BR) ($\mu + \sigma$)	54.69%± 0.0803	12.15%
W2A2 (within BR) (best)	54.86%	12.15%
W3A3	67.56%	–
W3A3 Random ($\mu + \sigma$)	66.72%±0.0441	7.94%
W3A3 Random (best)	66.78%	7.94%
W3A3 (within BR) ($\mu + \sigma$)	67.67%±0.0923	7.94%
W3A3 (within BR) (best)	67.83%	7.94%
W4A4	72.58%	–
W4A4 Random ($\mu + \sigma$)	72.35%±0.0617	3.61%
W4A4 Random (best)	72.422%	3.61%
W4A4 (within BR) ($\mu + \sigma$)	72.49%±0.0499	3.61%
W4A4 (within BR) (best)	72.546%	3.61%

DeiT-T. Consider $\mathrm{BR}_{0.005}$ as an example. We first inject random noise to weights within $\mathrm{BR}_{0.005}$ and train the model for one epoch. The results in Table 1 demonstrated that, surprisingly, randomly injecting noise to weights within $\mathrm{BR}_{0.005}$ does not lead to any accuracy degradation on 2-bit and 3-bit models but even improves the model accuracy in the best cases, suggesting that the final model is just a random *snapshot* of the model in oscillation, and there are abundant better solutions near the weight oscillation region.

Further, to examine if arbitrarily injecting noises to all the weights will have the same effect, we inject random noise to the same proportion of the total weights uniformly, instead of just weights within the BR. Predictably, this setting results in a severe accuracy drop, specifically in 2-bit and 3-bit settings. We also observe that there exist fewer weights in the oscillation boundary for models with higher bit-width, and that randomly injecting noise to weights either within $\mathrm{BR}_{0.005}$ or at arbitrary positions would result in a slight precision drop on the 4-bit DeiT-T, implying that the oscillation severity is anti-correlated to the bit-width. Based on the above discussion, we can confidently conclude that those oscillating weights are the primary factor that keeps the model from converging to good local optima.

4.2.3. Composition of the Oscillating Weights

Out of curiosity, we wanted to understand the composition of the oscillating weights, *i.e.,* whether some specific positions in a weight tensor are prone to vibration or oscillation

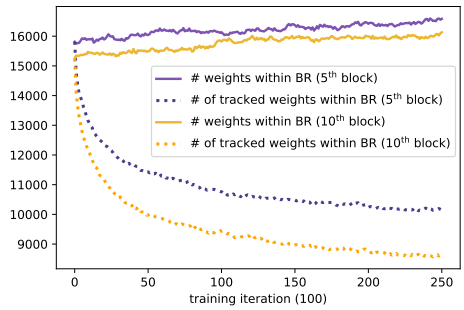

Figure 5: Comparison between the number of weights that is initially within the boundary range $\mathrm{BR}_{0.005}$ remaining in the boundary, and the total number of weights within $\mathrm{BR}_{0.005}$ of the $5^{\mathrm{th}}/10^{\mathrm{th}}$ transformer block in a 2-bit DeiT-T towards the end of the training.

weights are position agnostic. Following the rationale of our previous settings, we tracked weights that were initially within $\mathrm{BR}_{0.005}$ in the last few epochs in DeiT-T training. In Fig. 5, we notice that the number of tracked weights which continue to stay within $\mathrm{BR}_{0.005}$ gradually declines, while the total number of weights within $\mathrm{BR}_{0.005}$ is not reduced. This implies that although weights initially in the oscillation region gradually escape during optimization, new weights also enter the boundary region, preventing the network from getting rid of oscillation. This observation also suggests that the oscillating weights are position agnostic, and if we can prevent new weights from entering the boundary range, the oscillation will subside once all the weights are optimized to be outside the oscillation region. This naturally leads to our confidence-guided annealing solution in Sec. 5.2.

4.2.4. Bottleneck of Quantized ViTs

Previous literature has observed that quantizing self-attention operations brings the worst accuracy drop compared to other modules(Li & Gu, 2022; Li et al., 2022a;b). However, none of those works provides a concrete explanation for that. In the investigation, we found the most quantization-sensitive part of the self-attention lies in the multiplication between quantized *query* ($\mathbf{X}_{\mathcal{Q}_q}$) and *key* ($\mathbf{X}_{\mathcal{K}_q}$):

$$\mathbf{X}_{\mathcal{Q}_q} \cdot \mathbf{X}_{\mathcal{K}_q}^{\mathrm{T}} = F_q(F_q(\mathbf{X}) \cdot F_q(\mathbf{W}_{\mathcal{Q}}^{\mathrm{T}})) \cdot F_q(F_q(\mathbf{W}_{\mathcal{K}}) \cdot F_q(\mathbf{X}^{\mathrm{T}})) \quad (5)$$

Here F_q denotes the quantization function, and $\mathbf{X}$ denotes the input to the attention block. The corresponding back-propagation is formulated as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{\mathcal{K}}} \overset{STE}{\approx} \frac{\partial \mathcal{L}}{\partial \mathbf{X}_{out}} \cdot F_q(F_q(\mathbf{X}) \cdot F_q(\mathbf{W}_{\mathcal{Q}}^{\mathrm{T}})) \cdot F_q(\mathbf{X}^{\mathrm{T}}) \quad (6)$$

where $\mathbf{X}_{out} = \mathbf{X}_{\mathcal{Q}_q} \cdot \mathbf{X}_{\mathcal{K}_q}$. For simplicity, the terms $\mathbf{1}_{Q_n \leq F_q(\mathbf{W}_{\mathcal{K}}) \cdot F_q(\mathbf{X}^{\mathrm{T}})/\alpha \leq Q_p}$ and $\mathbf{1}_{Q_n \leq \mathbf{W}_{\mathcal{K}}/\alpha \leq Q_p}$ are omit-

ted here. The gradient *w.r.t.* $\mathbf{W}_{\mathcal{K}}$ depends on $F_q(F_q(\mathbf{X}) \cdot F_q(\mathbf{W}_{\mathcal{Q}}^{\mathrm{T}}))$ and $F_q(\mathbf{X}^{\mathrm{T}})$. Thus, $\mathbf{W}_{\mathcal{K}}$ optimization would be affected by the inaccurate estimation from oscillation in $F_q(\mathbf{W}_{\mathcal{Q}})$, and the resulting fluctuation in $F_q(\mathbf{W}_{\mathcal{K}})$ will, in turn, aggravate $F_q(\mathbf{W}_{\mathcal{Q}})$ oscillation. To decouple this oscillation aggravation loop, we propose re-parameterizing quantized *query-key* multiplication in Sec. 5.3.

5. Conquering Oscillation in Quantized ViT

We propose three novel techniques to abate quantization oscillations based on the above observations.

5.1. Statistical Weight Quantization

To eliminate the disproportionate influence from *outlier* weights to learnable scaling factor and reduce the oscillation interweave in between weights and scales, we propose statistic-based weight quantization StatsQ:

$$\mathbf{W}_q = \alpha_s \cdot ((\left\lfloor \mathrm{Clip}(\frac{\mathbf{W}}{\alpha_s}, -1, 1) \cdot n - 0.5 \right\rceil + 0.5) \cdot \frac{1}{n}) \quad \text{where } \alpha_s = 2 \cdot \frac{||\mathbf{W}||_{l1}}{|\mathbf{W}|},\ n = 2^{b-1} \tag{7}$$

Inspired by (Liu et al., 2022), α_s is calculated based on the maximum information entropy theory that when the quantized weights are evenly distributed in quantized levels, the highest entropy is preserved. Furthermore, the Clip function disregards the *outliers* in scaled weights, and the statistical calculation of scaling factor α_s equally counts the contribution from each weight in the update.

We further visualize the statistical scaling factor α_s throughout training in Fig. 6. In comparison to Fig. 2, we clearly see that α_s presents a much smoother progression and is almost steady toward the end of the training, demonstrating fewer fluctuations compared to the learnable scaling factors, and improves the overall QAT stability.

5.2. Confidence-Guided Annealing

Calming oscillations requires weights outside the volatility region to stop entering while optimizing out the weights that oscillate initially. Therefore, we propose confidence-guided annealing (CGA), which fine-tunes the final model by freezing weights outside the oscillation region and only updating the weights inside. The weight freezing happens every iteration, and once the weights escape the oscillation region, they are frozen, *i.e.*, not updated anymore. CGA is employed for n fine-tuning epochs following the completion of the regular training phase. The detailed implementation of CGA is summarized in Alg. 1.

The intuition of CGA also aligns with the weight confidence proposition in (Helwegen et al., 2019; Liu et al., 2021b): the confidence level of a quantized weight is proportional to the distance of its real-valued latent weights to the closest threshold. Weights far from the quantization threshold are considered as possessing "*high confidence*", while those oscillating weights close to the threshold are in "*low confidence*". In this sense, CGA freezes the "*high confident*" weights so that the "*less confident*" weights can be optimized with those fixed anchors as a prerequisite and gradually get out of the oscillation boundary. In comparison, the $\mathrm{iterative\ freezing}$ method proposed by (Nagel et al., 2022) freezes oscillating weights instead of the non-oscillating weights. We have tried $\mathrm{iterative\ freezing}$ on quantized DeiT-T and do not see any significant accuracy improvement. Thus, we argue that "*high confident*" weights should be frozen before the "*low confident*" weights, not the other way around, to facilitate better update direction estimation for the oscillating weights.

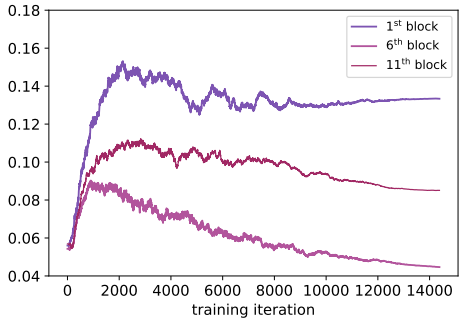

Figure 6: Trajectory of three statistical scaling factors α_s from the $1^{\mathrm{st}}/6^{\mathrm{th}}/11^{\mathrm{th}}$ transformer blocks in a 2-bit DeiT-T throughout training. The y-axis represents the value of α_s.

To further validate this point, we visualize the trajectories of two scaling factors α_{s} from the $1^{\mathrm{st}}/8^{\mathrm{th}}$ transformer block in a 2-bit DeiT-T throughout the confidence-guided annealing period. Fig. 7 depicts that the fluctuation of the scaling factor α_{s} ceases after a certain number of iterations. Since α_{s} is calculated with the statistics of the entire weight tensor, the cooling down of α_{s} reflects that all weights have successfully escaped the boundary range and are frozen, resulting in an *oscillation-free* model. Moreover, the top-1 accuracy of a 2-bit quantized DeiT-T is improved from 62.17% to 64.33% on ImageNet 1K by applying the CGA, which demonstrates the effectiveness of the proposed method.

5.3. *Query-Key* Reparametrization

Finally, we propose *query-key* reparameterization (QKR) to decouple the negative mutual-influence between quantized *query* and *key* weights ($\mathbf{W}_{\mathcal{Q}_q}$, $\mathbf{W}_{\mathcal{K}_q}$) oscillation. Simple but non-trivially, we reorder the multiplication of $\mathbf{X}_{\mathcal{Q}}$ and

Algorithm 1 Confidence-Guided Annealing

Given: Weight Matrix $\mathbf{W}$, Boundary Range $\mathrm{BR_x}$, Annealing Iterations n
for $j = 1$ **to** n **do**
  Calculate gradient $\mathbf{G}_j = \frac{\partial \mathcal{L}_j}{\partial \mathbf{W}}$
  $\mathbf{G}'_j = \mathbf{G}_j \cdot \mathbf{1}_{\widetilde{\mathbf{W}}_q \in \mathrm{BR_x}}$
    where $\widetilde{\mathbf{W}}_q = \mathrm{Clip}(\frac{\mathbf{W}}{\alpha_s}, -1, 1) \cdot n - 0.5$
  Update $\mathbf{W}$ with $\mathbf{G}'_j$
end for

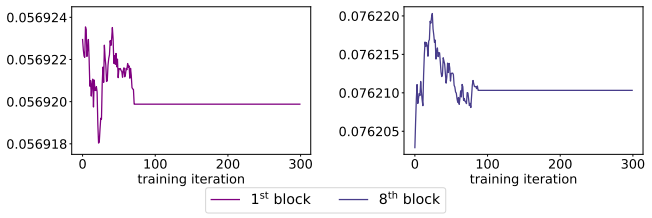

Figure 7: Trajectory of statistical scaling factors from the $1^{\mathrm{st}}/8^{\mathrm{th}}$ transformer block in a 2-bit DeiT-T throughout the Confidence-Guided Annealing training period. The y-axis represents the value of α_s. After a certain number of iterations, statistical scaling factors stop fluctuating, suggesting all weights have escaped the boundary range and are frozen.

$\mathbf{X}_\mathcal{K}$ in a self-attention module as follows:

$$\mathbf{X}_{out} = \mathbf{X}_\mathcal{Q} \cdot \mathbf{X}_\mathcal{K}^{\mathrm{T}} = \mathbf{X} \cdot (\mathbf{W}_\mathcal{Q}^{\mathrm{T}} \cdot \mathbf{W}_\mathcal{K}) \cdot \mathbf{X}^{\mathrm{T}} \tag{8}$$

Then with the multiplication reordering, the quantization can be formulated as follows:

$$\mathbf{X}_{out} = F_q(\mathbf{X}) \cdot F_q(F_q(\mathbf{W}_\mathcal{Q}^{\mathrm{T}} \cdot \mathbf{W}_\mathcal{K}) \cdot F_q(\mathbf{X}^{\mathrm{T}})) \tag{9}$$

Here F_q denotes the quantization function. The corresponding back-propagation is formulated as:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_\mathcal{K}} \overset{STE}{\approx} \frac{\partial \mathcal{L}}{\partial \mathbf{X}_{out}} \cdot F_q(\mathbf{X}) \cdot \mathbf{W}_\mathcal{Q}^{\mathrm{T}} \cdot F_q(\mathbf{X}^{\mathrm{T}}) \tag{10}$$

Similarly, the terms $\mathbf{1}_{Q_n \leq F_q(\mathbf{W}_\mathcal{Q}^{\mathrm{T}} \cdot \mathbf{W}_\mathcal{K}) \cdot F_q(\mathbf{X}^{\mathrm{T}})/\alpha \leq Q_p}$ and $\mathbf{1}_{Q_n \leq \mathbf{W}_\mathcal{Q}^{\mathrm{T}} \mathbf{W}_\mathcal{K}/\alpha \leq Q_p}$ are omitted here for simplicity. Compared to Eq. 6, in this new gradient derivation *w.r.t.* $\mathbf{W}_\mathcal{K}$, the quantized query weight $F_q(\mathbf{W}_\mathcal{Q})$ no longer exists such that the weight co-oscillation in the quantized *query-key* multiplication layers is decoupled. Besides, the proposed *query-key* reparameterization reduces the number of quantization operations from originally six in Eq. 5 to four in Eq. 9, which lessens the information loss resulting from discretization in the forward pass and estimates more accurate gradients in the backward pass.

6. Experiments

In this section, we evaluate our proposed methods on the DeiT-tiny, DeiT-small (Touvron et al., 2021) and Swin-tiny (Liu et al., 2021a) architectures on ILSVRC12 ImageNet classification dataset (Krizhevsky et al., 2017).

6.1. Implementation Details

We utilize the official implementation of DeiT[†] and Swin[§] Our quantized models are trained for 300 epochs with knowledge distillation using the corresponding full-precision models as the teacher models and as initialization. For quantized DeiT-T, 2-bit DeiT-S and 2-bit Swin-T, the training setting follows that of DeiT (Touvron et al., 2021) while without mixup/cutmix (Zhang et al., 2018b; Yun et al., 2019) data augmentation. For 3-bit/4-bit quantized DeiT-S and Swin-T, we follow the training recipe in (Li et al., 2022a;b). The number of annealing epochs is set to 25 for fine-tuning the optimized model with CGA. We apply 8-bit quantization for the first (patch embedding) layer and the last (classification and distillation) layers following (Esser et al., 2020; Li et al., 2022a;b).

6.2. Main Results

We name the combination of our methods ***O**scillation-**F**ree **Q**uantization* (OFQ) and present the overall performance on DeiT-T, DeiT-S and Swin-T and compare the result with baseline LSQ (Esser et al., 2020), Mix-QViT (Li et al., 2022b) which proposed a mix-precision quantized ViT, and QViT[‡] (Li et al., 2022a) in 2/3/4-bit quantization.

From Table 2, we can first observe that the performance degradation is more severe in lower bit quantization, that is, from real value to 2-bit quantization, the accuracy decreases by 17.57%, 11.9% and 10.8%, respectively on DeiT-T, DeiT-S and Swin-T using the baseline LSQ quantization. In addition, quantized DeiT-T suffers a greater performance drop compared to quantized DeiT-S and Swin-T. The above two observations are consistent with (Nagel et al., 2022)'s finding that models of smaller sizes and with lower bit-width exhibit a more severe accuracy drop due to oscillation. In comparison, the proposed OFQ eliminates the oscillation in the final model and thus substantially improves previous SoTAs and narrows the accuracy gap between the quantized model and their full-precision counterparts. Specifically, the 2-bit DeiT-S/Swin-T quantized with OFQ achieves 7.05%/4.64% higher accuracy than the previous state-of-the-art QViT (Li et al., 2022a), reducing the gap with real-valued models to only 4.18%/2.68%. Similarly, the 3-bit OFQ DeiT-T/DeiT-

[†] https://github.com/facebookresearch/deit

[§] https://pytorch.org/vision/0.13/models/swin_transformer.html

[‡] We have confirmed with the authors of (Li et al., 2022a) that their implementation could not establish Eq. 3 due to the reason discussed in Sec. 3.2. Therefore, we fixed their implementation, reran their experiments following their settings, and reported the results as QViT* in Table 2.

Table 2: Comparison of the proposed OFQ to previous ViT quantization methods on the ImageNet-1K dataset.

Network	Method	Bit-width	Top-1 %
DeiT-T	Full Precision	W32A32	72.02
	LSQ (Esser et al., 2020)	W2A2	54.45
	QViT* (Li et al., 2022a)	W2A2	50.37
	OFQ (Ours)	**W2A2**	**64.33**
	LSQ (Esser et al., 2020)	W3A3	68.09
	Mix-QViT (Li et al., 2022b)	∼W3A3	69.62
	QViT* (Li et al., 2022a)	W3A3	67.12
	OFQ (Ours)	**W3A3**	**72.72**
	LSQ (Esser et al., 2020)	W4A4	72.46
	Mix-QViT (Li et al., 2022b)	∼W4A4	72.79
	QViT* (Li et al., 2022a)	W4A4	71.63
	OFQ (Ours)	**W4A4**	**75.46**
DeiT-S	Full Precision	W32A32	79.9
	LSQ (Esser et al., 2020)	W2A2	68
	QViT* (Li et al., 2022a)	W2A2	68.67
	OFQ (Ours)	**W2A2**	**75.72**
	LSQ (Esser et al., 2020)	W3A3	77.76
	Mix-QViT (Li et al., 2022b)	∼W3A3	78.08
	QViT* (Li et al., 2022a)	W3A3	78.45
	OFQ (Ours)	**W3A3**	**79.57**
	LSQ (Esser et al., 2020)	W4A4	79.66
	Mix-QViT (Li et al., 2022b)	∼W4A4	80.11
	QViT* (Li et al., 2022a)	W4A4	80.33
	OFQ (Ours)	**W4A4**	**81.10**
Swin-T	Full Precision	W32A32	81.2
	LSQ (Esser et al., 2020)	W2A2	70.40
	QViT* (Li et al., 2022a)	W2A2	73.88
	OFQ (Ours)	**W2A2**	**78.52**
	LSQ (Esser et al., 2020)	W3A3	78.96
	Mix-QViT (Li et al., 2022b)	∼W3A3	79.45
	QViT* (Li et al., 2022a)	W3A3	80.06
	OFQ (Ours)	**W3A3**	**81.09**
	LSQ (Esser et al., 2020)	W4A4	80.47
	Mix-QViT (Li et al., 2022b)	∼W4A4	80.59
	QViT* (Li et al., 2022a)	W4A4	81.29
	OFQ (Ours)	**W4A4**	**81.88**

S/Swin-T significantly outperform the 3-bit models in QViT (Li et al., 2022a) and Mix-QViT (Li et al., 2022b).

In the 4-bit setting, we observe that LSQ can achieve comparable or even higher accuracy than full-precision counterparts, and most previous methods have already surpassed the full-precision models, which implies that the negative effect of oscillation in the 4-bit setting is less detrimental due to higher resolution of the model. Although the room for improvement becomes smaller, OFQ still consistently outperforms all previous works and the accuracy of full-precision models by 3.24%, 1.2%, and 0.68% on DeiT-T, DeiT-S, and Swin-T respectively.

Table 3: Ablation study on the individual effectiveness of the proposed statistical weight quantization (StatsQ), confidence-guided annealing (CGA), and QK reparameterization (QKR) on a quantized DeiT-S.

Network	Method	Bit-width	Top-1 %
DeiT-S	Baseline (LSQ)	W2A2	68
	StatsQ	W2A2	74.3
	StatsQ + QKR	W2A2	75.00
	StatsQ + CGA	W2A2	75.07
	StatsQ + QKR + CGA	W2A2	**75.72**
	Baseline (LSQ)	W3A3	77.76
	StatsQ	W3A3	78.56
	StatsQ + QKR	W3A3	79.15
	StatsQ + CGA	W3A3	79
	StatsQ + QKR + CGA	W3A3	**79.57**
	Baseline (LSQ)	W4A4	79.66
	StatsQ	W4A4	80.68
	StatsQ + QKR	W4A4	81.00
	StatsQ + CGA	W4A4	80.73
	StatsQ + QKR + CGA	W4A4	**81.10**

Table 4: Ablation study on the selection of boundary ranges ($\mathrm{BR_i}$) when applying CGA. The experiments are conducted using a 2-bit quantized DeiT-S.

Network	Method	Bit-width	Top-1 %
DeiT-S	without CGA	W2A2	75
	CGA ($\mathrm{BR}_{0.003}$)	W2A2	75.57
	CGA ($\mathrm{BR}_{0.005}$)	W2A2	**75.72**
	CGA ($\mathrm{BR}_{0.007}$)	W2A2	75.63
	CGA ($\mathrm{BR}_{0.01}$)	W2A2	75.67

6.3. Ablation Study

In this section, we first examine the solitary effectiveness of each of the three proposed methods (*i.e.*, StatsQ, CGA, and QKR) on DeiT-S quantization. As shown in Table 3, simply replacing (Esser et al., 2020) with StatsQ improves the accuracy by 6.3%, 0.8% and 1.0% in 2/3/4-bit settings, respectively. Moreover, QKR and CGA, when applied with StatsQ, can provide non-negligible improvement across bit-widths. For example, adding QKR improves 0.7% / 0.59% and adding CGA improves 0.77% / 0.44% in 2-bit / 3-bit settings. All three methods work collaboratively, and the final model with a combination of three methods boosts the accuracy by 7.72% on 2-bit DeiT-S compared to the baseline LSQ.

We then investigate the robustness of boundary range ($\mathrm{BR_x}$) selection in CGA on 2-bit quantized DeiT-S with 4 different settings $[\mathrm{BR}_{0.003}, \mathrm{BR}_{0.005}, \mathrm{BR}_{0.007}, \mathrm{BR}_{0.01}]$. In Table 4, we observe that $\mathrm{BR}_{0.005}$ brings the biggest accuracy improvement of 0.72% while $\mathrm{BR}_{0.003}$, $\mathrm{BR}_{0.007}$ and $\mathrm{BR}_{0.01}$ improve the accuracy by $\sim$ 0.6%. We further plot the trajectories of the statistical scaling factor with different BR settings, and Fig 8 illustrates that the larger the boundary

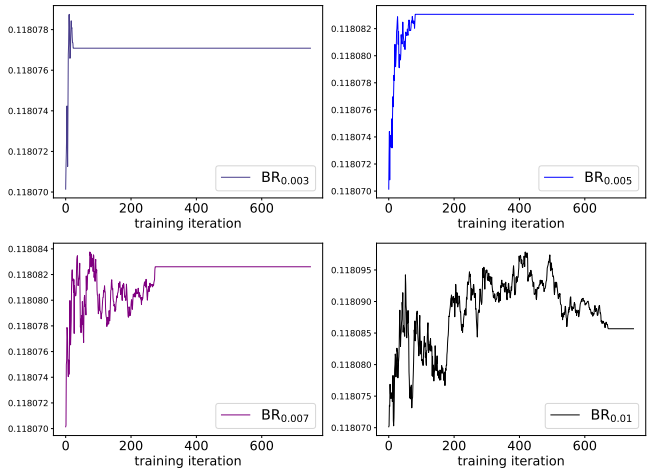

Figure 8: Trajectory of the statistical scaling factors from the 10th transformer block in a 2-bit DeiT-S throughout the confidence-guided annealing (CGA) period with 4 different boundary range settings ([$\mathrm{BR}_{0.003}$, $\mathrm{BR}_{0.005}$, $\mathrm{BR}_{0.007}$, $\mathrm{BR}_{0.01}$]). The y-axis represents the value of α_s. Refer to Appendix. 9.1 for the visualization of 3/4-bit DeiT-S.

range is, the more training iterations are required for the oscillating weights to exit the boundary range completely. The result implies that if $\mathrm{BR_x}$ is set too capacious, it will incur longer optimization time and would raise the risk of affecting "*high confident*" weights; on the contrary, if $\mathrm{BR_x}$ is too narrow, part of the "*low confident*" weights will be frozen, which restricts the effectiveness of CGA on updating oscillating weights. We empirically set the boundary range approximated to the average gradient in the optimized model, and Table 4 shows that CGA is pretty robust to the $\mathrm{BR_x}$ selections as long as $\mathrm{BR_x}$ is not set too aggressively.

7. Conclusion

In this work, we use Vision Transformer (ViT) as a study case for investigating quantization oscillation. We uncover the negative influence of learnable scaling factors on escalating weight oscillation. Moreover, we find that the quantized *query* and *key* of self-attention in ViT has a negative mutual influence that intensifies the weight oscillation collectively. In light of our observations, we introduce statistical weight quantization (StatsQ), confidence-guided annealing (CGA), and *query-key* reparameterization (QKR) to mitigate the oscillation of quantized weights. With StatsQ and QKR, quantization-aware training becomes more stable and converges to a better local minima; with CGA, oscillation is further subsided towards the end of the training and ultimately arrives at an oscillation-free model. For future work, we wish to investigate the oscillation phenomenon in more variety of DNNs on diverse applications to examine the effectiveness and generalizability of the proposed methods.

8. Acknowledgements

This research was supported by ACCESS - AI Chip Center for Emerging Smart Systems, sponsored by InnoHK funding, Hong Kong SAR, and HKSAR RGC General Research Fund (GRF) No.16203319.

References

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

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

Choi, J., Wang, Z., Venkataramani, S., Chuang, P. I.-J., Srinivasan, V., and Gopalakrishnan, K. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Esser, S. K., McKinstry, J. L., Bablani, D., Appuswamy, R., and Modha, D. S. Learned step size quantization. In *International Conference on Learning Representations*, 2020.

Gong, R., Liu, X., Jiang, S., Li, T., Hu, P., Lin, J., Yu, F., and Yan, J. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4852–4861, 2019.

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.

Helwegen, K., Widdicombe, J., Geiger, L., Liu, Z., Cheng, K.-T., and Nusselder, R. Latent weights do not exist: Rethinking binarized neural network optimization. *Advances in neural information processing systems*, 32, 2019.

Kenton, J. D. M.-W. C. and Toutanova, L. K. Bert: Pre-training of deep bidirectional transformers for language

understanding. In *Proceedings of naacL-HLT*, pp. 4171–4186, 2019.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, may 2017. ISSN 0001-0782. doi: 10.1145/3065386. URL https://doi.org/10.1145/3065386.

Li, Y., Dong, X., and Wang, W. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. In *International Conference on Learning Representations*, 2020.

Li, Y., Xu, S., Zhang, B., Cao, X., Gao, P., and Guo, G. Q-vit: Accurate and fully quantized low-bit vision transformer. In *Advances in Neural Information Processing Systems*, 2022a.

Li, Z. and Gu, Q. I-vit: integer-only quantization for efficient vision transformer inference. *arXiv preprint arXiv:2207.01405*, 2022.

Li, Z., Yang, T., Wang, P., and Cheng, J. Q-vit: Fully differentiable quantization for vision transformer. *arXiv preprint arXiv:2201.07703*, 2022b.

Liu, Z., Wu, B., Luo, W., Yang, X., Liu, W., and Cheng, K.-T. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 722–737, 2018.

Liu, Z., Mu, H., Zhang, X., Guo, Z., Yang, X., Cheng, K.-T., and Sun, J. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3296–3305, 2019.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021a.

Liu, Z., Shen, Z., Li, S., Helwegen, K., Huang, D., and Cheng, K.-T. How do adam and training strategies help bnns optimization. In *International Conference on Machine Learning*, pp. 6936–6946. PMLR, 2021b.

Liu, Z., Wang, Y., Han, K., Zhang, W., Ma, S., and Gao, W. Post-training quantization for vision transformer. *Advances in Neural Information Processing Systems*, 34: 28092–28103, 2021c.

Liu, Z., Cheng, K.-T., Huang, D., Xing, E. P., and Shen, Z. Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4942–4952, 2022.

Miyashita, D., Lee, E. H., and Murmann, B. Convolutional neural networks using logarithmic data representation. *arXiv preprint arXiv:1603.01025*, 2016.

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.

Nagel, M., Amjad, R. A., Van Baalen, M., Louizos, C., and Blankevoort, T. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, pp. 7197–7206. PMLR, 2020.

Nagel, M., Fournarakis, M., Bondarenko, Y., and Blankevoort, T. Overcoming oscillations in quantization-aware training. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16318–16330. PMLR, 17–23 Jul 2022.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wu, B., Dai, X., Zhang, P., Wang, Y., Sun, F., Wu, Y., Tian, Y., Vajda, P., Jia, Y., and Keutzer, K. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10734–10742, 2019.

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. Smoothquant: Accurate and efficient post-training quantization for large language models. *arXiv*, 2022.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Zhang, D., Yang, J., Ye, D., and Hua, G. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 365–382, 2018a.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018b.

Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., and Zou, Y. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

Zhu, S., Duong, L. H., and Liu, W. Xor-net: an efficient computation pipeline for binary neural network inference on edge devices. In *2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 124–131. IEEE, 2020.

9. Appendix

9.1. Progression of α_s in CGA with Different $\mathrm{BR_x}$

In this section, we provide the trajectory visualization of statistical scaling factors α_s from the 10th transformer blocks throughout confidence-guided annealing (CGA) period with different boundary ranges $[\mathrm{BR}_{0.003}, \mathrm{BR}_{0.005}, \mathrm{BR}_{0.007}, \mathrm{BR}_{0.01}]$ and compare their different behaviors in 2/3/4-bit DeiT-S. From Fig. 9-11, CGA demonstrates its capability of successfully guiding models to become oscillation-free regardless of model bit-widths and BR_x. Moreover, it is evident that more training iterations are required for all the weights to exit a larger boundary range across different bit-widths. Additionally, we can observe that fewer training iterations are needed for all the weights to stop oscillating in a 4-bit model than 2-bit/3-bit models, *e.g.*, $\sim$250 iterations for 4-bit DeiT-S, $\sim$1500 iterations for 3-bit DeiT-S and $\sim$700 iterations for 2-bit DeiT-S. This observation aligns with our findings that oscillation is less detrimental in the 4-bit setting due to the higher resolution of the model.

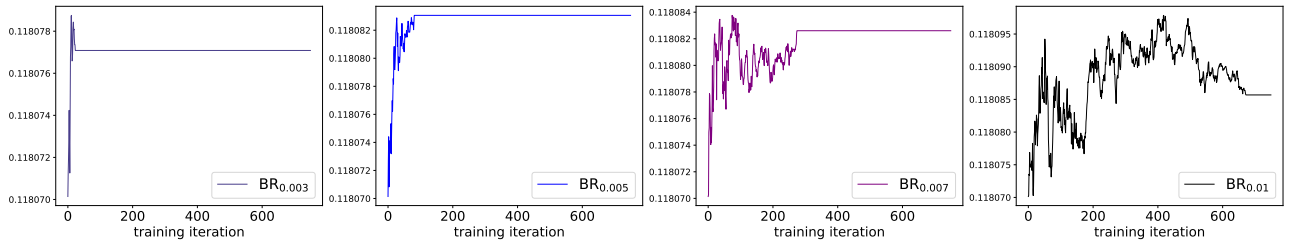

Figure 9: Trajectory of statistical scaling factors α_s from the 10th transformer blocks in a **2-bit** DeiT-S throughout CGA with 4 different boundary ranges $[\mathrm{BR}_{0.003}, \mathrm{BR}_{0.005}, \mathrm{BR}_{0.007}, \mathrm{BR}_{0.01}]$. The y-axis represents the value of α_s.

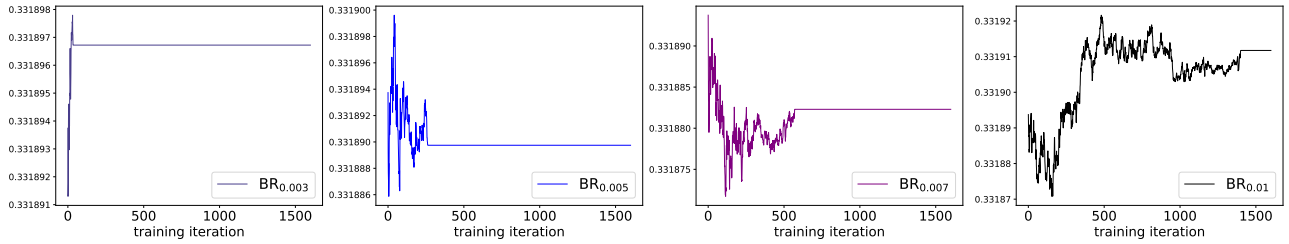

Figure 10: Trajectory of statistical scaling factors α_s from the 10th transformer blocks in a **3-bit** DeiT-S throughout CGA with 4 different boundary ranges $[\mathrm{BR}_{0.003}, \mathrm{BR}_{0.005}, \mathrm{BR}_{0.007}, \mathrm{BR}_{0.01}]$. The y-axis represents the value of α_s.

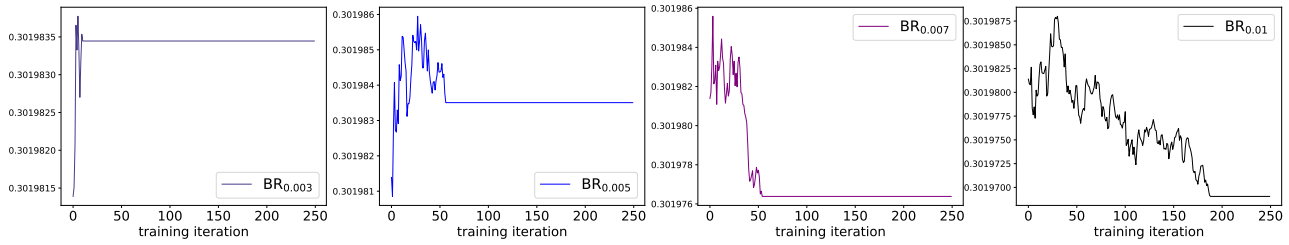

Figure 11: Trajectory of statistical scaling factors α_s from the 10th transformer blocks in a **4-bit** DeiT-S throughout CGA with 4 different boundary ranges $[\mathrm{BR}_{0.003}, \mathrm{BR}_{0.005}, \mathrm{BR}_{0.007}, \mathrm{BR}_{0.01}]$. The y-axis represents the value of α_s.