SUB-8-BIT QUANTIZATION FOR ON-DEVICE SPEECH RECOGNITION: A REGULARIZATION-FREE APPROACH

Kai Zhen, Martin Radfar, Hieu Nguyen, Grant P. Strimel, Nathan Susanj, Athanasios Mouchtaris

Amazon Alexa AI

ABSTRACT

For on-device automatic speech recognition (ASR), quantization aware training (QAT) is ubiquitous to achieve the trade-off between model predictive performance and efficiency. Among existing QAT methods, one major drawback is that the quantization centroids have to be predetermined and fixed. To overcome this limitation, we introduce a regularization-free, "soft-to-hard" compression mechanism with self-adjustable centroids in a μ-Law constrained space, resulting in a simpler yet more versatile quantization scheme, called General Quantizer (GQ). We apply GQ to ASR tasks using Recurrent Neural Network Transducer (RNN-T) and Conformer architectures on both LibriSpeech and de-identified far-field datasets. Without accuracy degradation, GQ can compress both RNN-T and Conformer into sub-8-bit, and for some RNN-T layers, to 1-bit for fast and accurate inference. We observe a 30.73% memory footprint saving and 31.75% user-perceived latency reduction compared to 8-bit QAT via physical device benchmarking.

***Index Terms*—** On-device speech recognition, quantization aware training, RNN-T, conformer, model efficiency

1. INTRODUCTION

Improving the efficiency of neural automatic speech recognition (ASR) models via quantization is critical for on-device deployment scenarios. For neural network accelerator (NNA) embedded devices, where memory and bandwidth are at a premium, quantization can reduce the footprint and lower the bandwidth consumption of ASR execution, which will not only afford a faster model inference but also facilitate model deployment to various portable devices where a stable network connection is limited.

Existing quantization methods can be post-training quantization (PTQ) or in-training / quantization aware training (QAT). PTQ is applied after the model training is complete by compressing models into 8-bit representations and is relatively well supported by various libraries [1, 2, 3, 4, 5, 6], such as TensorFlow Lite [7] and AIMET [8] for on-device deployment. However, almost no existing PTQ supports customized quantization configurations to compress machine learning (ML) layers and kernels into sub-8-bit (S8B) regimes [9]. Moreover, the performance drop is inevitable as the model is unaware of the loss of precision when being quantized at test time. In contrast, QAT performs bit-depth reduction of model weights (for example, from 32-bit floating point to 8-bit integer) during training which usually yields superior performance over PTQ [10][11]. The QAT mechanism can be in the forward pass (FP-QAT) or the backward pass (BP-QAT), with the difference being whether regularization is used in the loss function. FP-QAT [9] quantizes the model weights during forward propagation to pre-defined quantization centroids. BP-QAT [12, 13, 14] relies on customized regularizers to gradually force weights to those quantization centroids (i.e., "soft quantization" via gradient) during training before hard compression performs in the late training phase. As model weights are informed by the customized regularizers to move closer to where they are quantized at runtime per training step, the predictive performance is often well preserved. Therefore, the focus of this work is on QAT.

Under both FP- and BP-QAT, it is essential that the quantization centroids are defined and specified before model training. As such, the demerit is the low feasibility when quantizing models in S8B mode because one needs to select the proper quantization centroids and their configurations for each kernel in each layer to ensure minimal runtime performance degradation. Consequently, applying existing QAT methods to Conformer [15] becomes quite challenging, as it usually contains more than hundreds of kernels.

In this work, we propose General Quantizer (GQ), a regularization-free, model-agnostic quantization scheme with a mixed flavor of both FP- and BP-QAT. GQ is "general" in that it does not augment the objective function by introducing any regularizer as in BP-QAT but determines the appropriate quantization centroids during model training for a given bit depth, and it can be simply applied in a plug-and-play manner to an arbitrary ASR model. Unlike FP-QAT, GQ features a soft-to-hard quantization during training, allowing model weights to hop around adjacent partitions more easily. Under GQ, quantization centroids are self-adjustable but in a μ-Law constrained space. As a proof-of-concept, we adopt the ASR task and conduct experiments on both the LibriSpeech and de-identified far-field datasets to evaluate GQ on three major end-to-end ASR architectures, namely conventional Recurrent Neural Network Transducer (RNN-

978-1-6654-7189-3/22/$31.00

T) [16], Bifocal RNN-T [17], and Conformer [18][19]. Our results show that in all three architectures, GQ yields little to no accuracy loss when compressing models to S8B or even sub-5-bit (5-bit or lower). We also present performance optimization strategies from ablation studies on bit-allocation and quantization frequency. Our contributions are as follows:

- We propose GQ, inspired by both FP- and BP-QAT approaches. GQ enables on-centroid weight aggregation without augmented regularizers. Instead, it leverages Softmax annealing to impose soft-to-hard quantization on centroids from the μ-Law constrained space.
- GQ supports different quantization modes for a wide range of granularity: different bit depths can be specified for different kernels/layers/modules.
- With GQ, we losslessly compress a lightweight streaming Conformer into sub-5-bit with more than $6\times$ model size reduction. To our best knowledge, this is among the first sub-5-bit Conformer models for on-device ASR. Without accuracy degradation, our GQ-compressed 5-bit Bifocal RNN-T reduces the memory footprint by 30.73% and P90 user-perceived latency (UPL) by 31.30%.

We describe the problem in Sec. 2 and GQ in Sec. 3. The experimental settings and results are detailed in Sec. 4. We conclude in Sec. 5 with some final remarks.

2. PRELIMINARIES

2.1. Problem Formulation

Consider a general deep neural network architecture with K layers, $\mathcal{F} = \mathcal{F}_1 \circ \cdots \circ \mathcal{F}_K$, mapping the input from $\mathbb{R}^{d_1}$ to the output in $\mathbb{R}^{d_{K+1}}$ as $\mathcal{F} : \mathbb{R}^{d_1} \longmapsto \mathbb{R}^{d_{K+1}}$, where the input and output of an arbitrary k-th layer are $\boldsymbol{x}^{(k+1)} := \mathcal{F}_k(\boldsymbol{x}^{(k)})$. Under supervised learning, the training data $\mathcal{X} = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x}_N\}$ and $\mathcal{Y} = \{\boldsymbol{y_1}, \ldots, \boldsymbol{y}_N\}$ are used for updating model weights $\mathbb{W} = [\boldsymbol{W}_1, \ldots, \boldsymbol{W}_K]$ for K layers in $\mathcal{F}$. Usually the optimization process is over the training objective function $\mathcal{L}(\mathcal{X}, \mathcal{Y}, \mathcal{F}, \mathbb{W}) = \frac{1}{N} \sum_{i=1}^{N} \ell(\mathcal{F}(\boldsymbol{x_i}), \boldsymbol{y_i}) + \lambda R(\mathbb{W})$, where i is the data batch index, ℓ is the major loss term measuring model accuracy and $R(\mathbb{W})$ is the regularizer blended to the objective function via a coefficient λ.

Network quantization aims at discretizing model weights. For scalar quantization, it is to convert each weight, $w \in \mathbb{W}$, to a quantization centroid, $z \in \boldsymbol{z}$, where $\boldsymbol{z} = [z_1, ...z_m]$ to ensure the network is compressed into $\lceil log_2 m \rceil$-bit. For S8B quantization, centroids are from a subset of `INT8` values.

2.2. Related QAT Approaches

BP-QAT counters model weight continuity via regularization. For example, it introduces weight regularizers on model weights, i.e., $R(\mathbb{W})$, measuring the point-wise distance between each weight and m quantization centroids in the centroid vector $\boldsymbol{z} = [z_1, ...z_m]$. Note that the quantization weight regularizer in the loss function, as $R(\mathbb{W})$, must be gradient descent compatible. Consequently, $R(\mathbb{W})$ cannot enforce each weight to be replaced by the closest centroid in $\boldsymbol{z}$ as $w = \arg\min_i ||w - z_i||$, for $w \in \mathbb{W}$ and $z_i \in \boldsymbol{z}$, because the $\min$ operator is not differentiable. Recent BP-QAT methods force weights to approach the centroid in $\boldsymbol{z}$ using $R(\mathbb{W}) = \sum_{w \in \mathbb{W}} \mathcal{D}(w, \boldsymbol{z})$, where the differentiable dissimilarity function $\mathcal{D}$ is based on a cosine function in [12, 13].

In contrast, FP-QAT can be regularizer free [9, 20]. Usually, the process is to use a "fake quantizer" or equivalent operations during training, hard quantizing weights to a specific range and bit-depth; and then at runtime, converting the model to `INT8` format via TFLite [21]. The study [9] uses native quantization operators with which, during training, the weights are quantized and then converted to the integer type for model deployment. However, FP-QAT is essentially hard compression recurring during training with severely dropped performance when applied to S8B quantization. Consequently, finetuning is usually needed, which prolongs the model training time [22, 23].

Both FP-QAT and BP-QAT require specifying appropriate quantization centroids before model training. While the centroids for `INT8` model compression are pre-defined, for S8B quantization, the optimal set of centroids is usually kernel/layer specific. For models, such as Conformer [15], where there are usually hundreds of kernels, current S8B QAT methods become less tractable.

In this work, we combine the merit from both FP- and BP-QAT and propose General Quantizer (GQ) that navigates weights to the corresponding quantization centroids without introducing augmented regularizers but via feedforward-only operators. Our work is inspired by a continuous relaxation of quantization [24] also used for speech representation learning [25, 26, 27, 28, 29], and μ-Law algorithm for 8-bit pulse-code modulation (PCM) digital telecommunication [30].

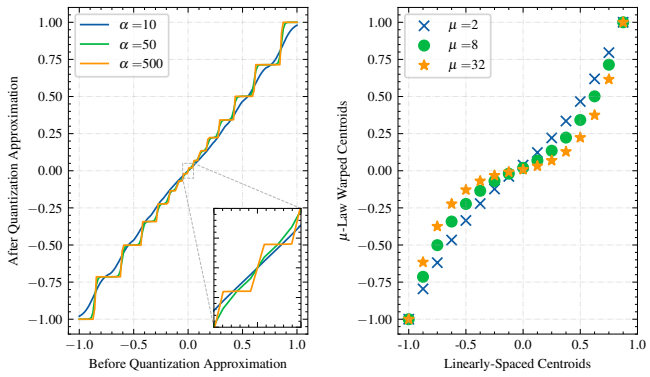
3. METHODS

3.1. Centroid Selection via Softmax-Based Dissimilarity Matrices

For any weight value $w_i \in \boldsymbol{w}$ where $|\boldsymbol{w}| = n$, and the quantization centroid vector $\boldsymbol{z} = [z_1, ...z_m]$, we define the point-wise dissimilarity matrix in Eq. 1

$$\boldsymbol{A}_{\text{soft}} = \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{bmatrix}, \tag{1}$$

where a_{ij} is the probability of representing w_i by z_j. Each row in $\boldsymbol{A}_{\text{soft}}[i\cdot]$ is summed to 1 with the largest probability



(a) Weight value before and after quantization approximation. Here, the larger the softmax temperature α is, the closer to true quantization that the weight transformation becomes.

(b) Relationship between the wrapped quantization centroids and μ. A large μ means more quantization centroids are allocated near 0 and vice versa.

Fig. 1: Hyperparameters in GQ: α adjusts the quantization temperature which increases gradually during training, and μ modulates the non-linearity of quantization centroids.

going to the closest centroid. This is achieved when the pointwise distance $||w_i - z_j||_1$ is scaled by a negative number $-\alpha$ and wrapped by a Softmax function in Eq. 2

$$a_{ij} = \frac{e^{-\alpha||w_i - z_j||_1}}{\sum_{j=1}^{m} e^{-\alpha||w_i - z_j||_1}}. \tag{2}$$

Here, $\alpha \in [1, \infty)$ serves as the Softmax temperature for quantization annealing. When α is relatively small, w_i will be approximated by all centroids in $\boldsymbol{z}$ (see Eq.3); when $\alpha \to \infty$, $\boldsymbol{A}_{\text{soft}}[i\cdot]$ becomes a one-hot vector $\boldsymbol{A}_{\text{hard}}[i\cdot]$ and the weight will be the closest centroid.

$$\overline{w_i} = \boldsymbol{z} \times \boldsymbol{A}_{\text{soft}}[i\cdot]^T. \tag{3}$$

For simplicity, during training, we set the initial and target scalar values to be α_{start} and α_{end}, and allow α to gradually and linearly increase from s_{start} to s_{end}, as shown in Eq. 4.

$$\alpha = \alpha_{\text{start}} + (s - s_{\text{start}}) \times \frac{\alpha_{\text{end}} - \alpha_{\text{start}}}{s_{\text{end}} - s_{\text{start}}}. \tag{4}$$

As a result, the QAT effect is gradually intensified. At $\alpha = 10$, a rather small value, weights after being approximated by quantization centroids in $\boldsymbol{z}$ roughly preserve their original values; however, as α gradually increased to 500, the near-linear line almost becomes a step function, aggregating weights to just a few centroids (see Fig. 1 (a)). This forms a soft-to-hard QAT and allows model weights to be updated via gradients with barely any extra constraint during the early stage of training before driving weights to a certain centroid.

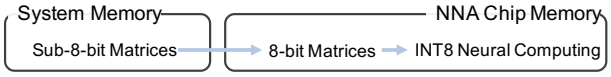


Fig. 2: Loading weights from system memory to chip memory on NNA is faster in S8B format, although the arithmetic operation is still in `INT8` format.

3.2. Adjusting Centroids with μ-Law Expanding

We assume weight distribution symmetry from any kernel in a trained 32-bit neural network in which the absolute values of most weights are small. Consequently, the imposed m quantization centroids in $\boldsymbol{z}$ should also be symmetric ($|z_i| = |z_{m+1-i}|$ where $i \in [1, m]$) with most centroids close to 0. To specify and adjust the level of non-linearity of $\boldsymbol{z}$ per kernel during training, we resort to μ-Law algorithm, mainly used in 8-bit PCM telecommunication (similar to A-Law algorithm standardized in Europe). The motivation for using μ-Law function is that it accents samplings from small (soft) values, reducing the quantization error and increasing signal-to-quantization-noise-error (SQNR) for data transmission. Hence, we employ the μ-Law algorithm in GQ to improve the quantization robustness of ASR models.

In μ-Law expanding function (Eq. 5), $\boldsymbol{z}$, linearly spaced values within the range of -1 and 1, are warped as $\boldsymbol{z'}$ in which the values are driven closer to 0, except for the boundary poles. As shown in Fig. 1 (b), when μ increases, the linearly spaced values are more noticeably warped in the μ-Law transformed space: by adjusting the value of μ that minimizes the quantization error $\arg\min_{\mu} ||\boldsymbol{z'} \times \boldsymbol{A}_{\text{soft}} - \boldsymbol{w}||_2$, quantization centroids in $\boldsymbol{z}$ can be re-distributed to better reflect the dynamic weight range of a specific neural component. A larger μ means the weight distribution is concentrated near 0; therefore, we allocate more quantization centroids near the origin. Smaller μ values indicate the weight distribution is tail-heavy.

$$\boldsymbol{z'} = \text{sng}(\boldsymbol{z}) \left(\frac{(1+\mu)^{|\boldsymbol{z}|} - 1}{1 + \mu} \right). \tag{5}$$

3.3. S8B Model for 8-Bit Computing

Due to limited chip memory size and bandwidth of the NNA, weights are loaded from system memory to the chip memory per matrix, which is time consuming. Hence, compressing the model into S8B can achieve inference speedup, even though NNA uses `INT8` for neural computing (see Fig.2).

Nonetheless, we map $\boldsymbol{z'}$ in Eq. 5 to the closest value in $[\cdots k/128 \cdots]$, where the integer $k \in [-128, 127]$, such that in-training and runtime quantization centroids are consistent.

3.4. Callback "Is All You Need"

A callback is a set of functions to be invoked at certain training stages. Under GQ, the callback is all you need: For any