ON-DEVICE CONSTRAINED SELF-SUPERVISED LEARNING FOR KEYWORD SPOTTING VIA QUANTIZATION AWARE PRE-TRAINING AND FINE-TUNING

Gene-Ping Yang[⋆]*, Yue Gu*[†]*, Sashank Macha*[†]*, Qingming Tang*[†]*, Yuzong Liu*[§]

[⋆]Centre for Speech Technology Research, University of Edinburgh
[†]Alexa Perceptual Technologies, Amazon [§]Zoom Video Communications, Inc.

ABSTRACT

Large self-supervised models have excelled in various speech processing tasks, but their deployment on resource-limited devices is often impractical due to their substantial memory footprint. Previous studies have demonstrated the effectiveness of self-supervised pre-training for keyword spotting, even with constrained model capacity. In our pursuit of maintaining high performance while minimizing the model's resource demands, we investigate the implementation of Quantization Aware Training for both self-supervised pre-training and fine-tuning, specifically tailored to fit within the constraints of on-device model budget. Our experiments emphasize the critical role of selecting and synchronizing QAT methods throughout both stages of model training and tuning. We evaluate our methodology on a 16.6k-hour in-house keyword spotting dataset, and show that there is no decline in performance, even when the bit size of model weights and activations is cut by a factor of four.

Index Terms— Self-supervised learning, keyword-spotting, on-device classification, quantization aware training, low-bit quantization

1. INTRODUCTION

Self-supervised speech representation learning (S3RL) has emerged as an exceptionally effective pre-training approach [1, 2, 3, 4, 5]. It enables the learning of speech features that are versatile across a range of tasks, including automatic speech recognition, speaker verification, emotion recognition, and keyword spotting [6, 7, 8, 9]. However, many S3RL models, often built as 12-layer or 24-layer Transformer encoders, demand substantial computational and memory resources. This makes them less suited for real-time on-device applications with limited resources. Given the rising demand for on-device speech processing, particularly in always-on, low-power voice assistants, downsizing S3RL models becomes essential for their practical deployment in real-world scenarios.

To address the above issue, previous research has explored self-supervised pre-training using models of reduced size for keyword spotting. Gao et al. [8] evaluated various self-supervised pre-training techniques on a compact 3-layer transformer. The authors demonstrated that models employing self-supervised pre-training consistently outperformed the models without S3RL, with the autoregressive predictive coding (APC) method [1, 5] proving to be the most effective, yielding the best results. In contrast, Yang et al. [9] developed an efficient and highly capable on-device constrained S3RL model, specializing in keyword spotting using knowledge distillation method. The process of distilling knowledge from the Wav2vec 2.0 [2] 12-layer teacher model into a 3-layer transformer, featuring reduced hidden dimensions, resulted in a significant enhancement of keyword spotting performance.

While both methods have made significant improvements in model performance with the constraints of model size using self-supervised techniques, our goal is to push the limit even further. Instead of solely reducing model depth and dimension as in previous methods, we pivot our attention to minimizing the bit size of model weights. Quantization aware training (QAT) is an appealing approach for reducing the bit size of the models [10, 11, 12, 13, 14]. It aims to replicate low-precision behavior throughout training to ultimately achieve a finely quantized model using significantly fewer bits, e.g. 32 bits to 8 bits. The QAT strategies achieve model compression by reducing the bit size utilized to represent both weights and activations [15, 16, 17, 18, 19]. A notable advantage of this approach lies in its ability to preserve the model's depth and width, which has been shown to capture unique information that proves advantageous for a range of tasks [20].

A common QAT approach uses dual-model training. One model operates with full precision, while the other, with low bit precision, is trained to mimic the teacher feature via knowledge distillation [10, 11, 21, 16, 22, 23]. Yeh et al. [24] have demonstrated that this approach effectively reduces model size while preserving performance in speech-related tasks. Another QAT technique, termed absolute cosine regularization (ACR) [13, 14], offers a more streamlined approach by eliminating the dual-model training, targeting on a single and efficient model training for QAT. This method incorporates an extra loss to ensure the model weights align with the intended quantized values.

In this paper, we focus on compressing and optimizing the on-device low-bit S3RL model using QAT, distinguishing our approach from previous research [8, 9]. To realize our goal, we assessed the role of ACR in self-supervised pre-training. In addition, we explore ACR in conjunction with activation quantization, employing either moving average quantization (***MA***) or dynamic quantization (***Dyn***). We apply these QAT methods to both self-supervised pre-training and fine-tuning for keyword spotting. Building upon prior studies that emphasize the advantages of S3RL in keyword spotting [8, 9], we evaluate our approach using 3-layer transformer while further reducing the bit size. Our findings indicate that employing synchronized Quantization Aware Training (QAT) for both self-supervised pre-training and fine-tuning maintains consistent performance, with no significant drop in comparison to a full-precision S3RL model, which is four times larger in terms of model weights and activations. Our major findings are as listed follow:

1. Incorporating QAT during the S3RL pre-training phase effectively bridges the performance gap observed in fine-tuning when compared to a full-precision S3RL model.
2. The application of ACR may impose overly restrictive con-

[⋆§]Work done at Amazon

straints, potentially resulting in undesired behaviors in model weights with low utilization.

3. The proposed QAT approach achieves an overall compression rate of 24.1% with consistency performance when considering model weight pruning, making it the most efficient method in terms of quantization value utilization among all QAT techniques.

2. METHODOLOGY

In this section, we introduce different quantization aware training (QAT) strategies and the self-supervised pre-training approach. Following the method outlined by Gao et al. [8], we investigate the impact of autoregressive predictive coding (APC) in a low-bit scenario.

2.1. Autoregressive Predictive Coding

Among the various self-supervised pre-training approaches tested in prior work, including APC [1], contrastive predictive coding (CPC) [25], and masked predictive coding (MPC) [26], it was found that APC performed the best, as reported by Gao et al. [8]. The aim of APC is to predict future frames based on past information, operating on the premise that a model capable of understanding the data can make predictions about the future [5]. Given an sequence of log-filterbank energy (LFBE) $x_1, x_2, ..., x_T$, the objective is to predict the frame k steps into the future x_{t+k} given the frames up to the current one $x_{1:t}$. Formally, the objective function is

$$L_{APC} = \sum_{t=1}^{T-k} \|x_{t+k} - f_W(x_{1:t})\|_2^2, \tag{1}$$

where f_W is a 3-layer transformer and W is the model weights.

2.2. Absolute Cosine Regularization

Absolute Cosine Regularization (ACR) [13], a Quantization Aware Training (QAT) method applied to model weights, introduces a regularization loss that encourages the weights to approach specific predefined quantized values. It is designed to work in conjunction with a linear quantization scheme (e.g., int8 representation), where the quantized values are uniformly distributed. In this context, the absolute cosine function serves as a means to assess the proximity of each model weight to one of the quantized values. Given an absolute cosine function $g(x) = |cos(\pi f x)|$, the peak value of 1 occurs when $x \in \{..., -\frac{3}{f}, -\frac{2}{f}, -\frac{1}{f}, 0, \frac{1}{f}, \frac{2}{f}, \frac{3}{f}, ...\}$. The objective is to compute a score $g(w_i)$ for each model weight w_i, with a higher score indicating a better match between the model weights and these quantized values. The ACR loss function is then defined as follows:

$$L_{ACR} = -\sum_i |cos(\pi f w_i)|. \tag{2}$$

Here, w_i represents the i^{th} scalar weight in the model, and f is a frequency parameter that determines the bit size according to a specific linear quantization scheme. For instance, when working with a bit size of 8 and weights ranging from [-1, 1), we can set $f = 128$. This results in the $g(w_i)$ score peaking at values such as $-1, -\frac{127}{128}, -\frac{126}{128}, ..., \frac{126}{128}, \frac{127}{128}$.

The overall objective of training the APC model with ACR is as follows:

$$L_{total} = L_{APC} + \alpha L_{ACR}, \tag{3}$$

where α is a hyperparameter that controls strength of the regularization term. In summary, ACR offers a soft regularization approach that guides the model weights toward specific quantized values without requiring them to precisely match those values during training.

2.3. Moving Average Quantization

Besides model weight quantization, prior research has often employed hard quantization methods for quantizing activations [27, 28]. In hard quantization, each activation is mapped to a predefined set of quantization values, such as those within the range [-1, +1] or [0, +1]. However, this approach can be limiting because different layers may exhibit varying dynamic ranges for their activations. To address this constraint, we leverage the moving average quantization scheme (***MA***) for activations. This scheme dynamically adjusts the minimum and maximum of the quantization range based on the current batch's values, providing greater flexibility. We categorized the activations based on their usage, such as query, key, value, or the softmax operation, each having its unique range in different layer. This quantization scheme relies on two parameters: (1) n: the minimum of the range, and (2) m: the maximum. These parameters are updated during each training iteration as follows:

$$\mathrm{n}_t = \mathrm{n}_{t-1} \times 0.99 + \min(A_t) \times 0.01$$
$$\mathrm{m}_t = \mathrm{m}_{t-1} \times 0.99 + \max(A_t) \times 0.01$$

where $\mathrm{n}_t, \mathrm{m}_t$ indicated two values at iteration t, and A_t represents the activation values. Based on this range, at iteration t, the activations a_t are clipped to the closest value in the set $\{\mathrm{n}_t, \mathrm{n}_t + k, \mathrm{n}_t + 2k, ..., \mathrm{m}_t\}$, where $k = \frac{\mathrm{m}_t - \mathrm{n}_t}{q-1}$ when we allow q quantized values. Therefore, with just 2 additional parameters for each activation, we can dynamically allocate the range of each quantization set.

2.4. Dynamic Quantization

To further reduce the restriction on the dynamics of the activation, we leverage dynamic quantization (***Dyn***) by assigning n and m based on the activations of current training iteration and dividing each activation into additional groups. For an hidden activation of shape (f, d) where f represents frame index and d represents feature dimension, we assign n, m values independently to each f, allowing each frame in the activation to be quantized separately. We obtained n, m for each f at iteration t as follows:

$$\mathrm{n}_{t,f} = \min(\mathrm{A}_{t,f}), \quad \mathrm{m}_{t,f} = \max(\mathrm{A}_{t,f}) \tag{4}$$

This approach, commonly used in previous work, ensures more flexible quantized values [10, 11].

For both ***MA*** and ***Dyn***, we derive the quantized activation A^Q from A as follows:

$$A_{ij}^Q = \text{round}\left(\frac{A_{ij} - n}{m - n} \times (q-1)\right) \times \frac{m-n}{(q-1)} + n \tag{5}$$

To maintain gradient information during quantization-aware training, we employ the straight-through estimator technique [29] since the rounding operation itself does not provide gradient information.

3. DATASET AND IMPLEMENTATION

3.1. Dataset

Our experiments are carried out on an in-house keyword spotting dataset, which comprises 16.6k hours of de-identified audio recordings captured under various front-end conditions. All the data has

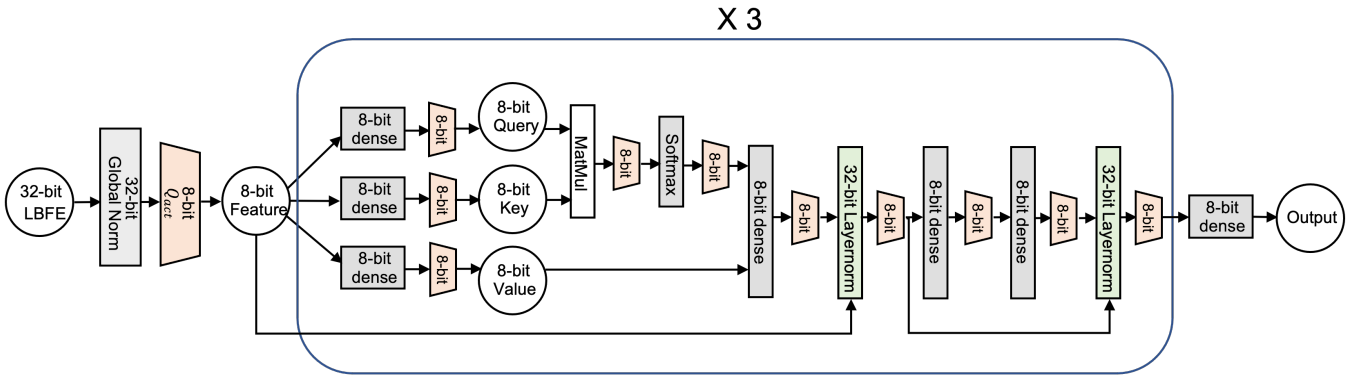

Fig. 1: Diagram of the quantization aware training model. The circle indicates input or hidden activations, the orange trapezoidal indicated the quantization operation on activations, and the grey block indicates the model weights with desired quantized bit width.

been pre-processed to 64-dimensional log-filterbank energy (LFBE) features, using a window size of 25ms and a shift of 10ms. Each sample is truncated to a length of 100 frames (equivalent to 1 second). The dataset is divided into 85 hours each for validation and testing, with the remaining data being used for training. The test set is further partitioned into two subsets: normal condition (consisting of clean speech) and playback condition (including increased noise). The labels for the dataset are based on human annotation.

3.2. Implementation

In line with prior studies [8, 9], we implement a 3-layer transformer model. This model consists 4 attention heads, a hidden dimension of 256, and a fully connected layer with 512 dimensions in each layer. For S3RL, we employ the Autoregressive Predictive Coding (APC) objective, which involved predicting 8 frames into the future with an 80 ms shift. During self-supervised pre-training, the model is trained with APC for 15 epochs, with each epoch comprising 5,000 steps and a batch size of 2,048. We utilize a learning rate of 0.001, a dropout rate of 0.1, and the Adam optimizer. During the fine-tuning phase for keyword spotting, we add an additional linear classifier on top of the hidden activation after the final transformer layer. The model undergo further fine-tuning for 30 epochs, maintaining the same steps and batch size per epoch as in the S3RL training.

We present an illustration of our Quantization Aware Training (QAT) S3RL model in Figure 1. In contrast to previous approaches applied to transformers [10, 11], where certain components such as the softmax, biases, and the last linear layer were left unquantized, our quantization strategy encompasses the quantization of inputs, all model weights, and activations, including the bias term, softmax operation, and the final task-specific linear layer. The only component left unquantized is the layernorm, which represents less than 0.1% of the total parameters.

After quantization, our model is reduced to an 8-bit size, resulting in a 75% reduction in the overall model size. For Absolute Cosine Regularization (ACR), we impose a constraint weight range between [-1, 1) and set the frequency parameter f in the absolute cosine function to 128. Regarding moving average activation quantization, we initialize the minimum and maximum values n_0 and m_0 for each hidden activation to [-6, 6], except for the input, which is set to [0, 32], and the softmax output, which is set to [0, 1]. Dynamic activation quantization does not require any specific configuration. It's worth noting that once quantization aware training is completed, post-training quantization on model weights becomes necessary, as ACR does not employ hard quantization during the forward pass.

3.3. Evaluation

To assess the effectiveness of our method, we evaluate the models using our in-house dataset. We measured the false acceptance rate (FAR) at a fixed false rejection rate (FRR) while keeping the FRR at the operating point (OP) of the baseline model. The FRR is the ratio of false negatives to true positives. We determined the OP at which our method exhibited a comparable FRR to that of the baseline model and used the same OP to compute the corresponding FAR, which represents the ratio of false positives to true negatives.

4. RESULTS

4.1. QAT vs. PTQ

We present our proposed Quantization-Aware Training (QAT) methods and several baseline models in Table 1. As a reference, we setup a full-precision (FP) S3RL baseline (1) following [8].

For the FP models with Post Training Quantization (PTQ) (2, 3), we quantized the model weights with a range of [-1, 1], and the activations were quantized using either moving average or dynamic quantization. In the case of w8a8$_{\text{MA}}$ (2), we froze the model weights and determined the moving average n, m of each activation group over 5,000 iterations. Our findings indicate that the integration of PTQ resulted in a degradation of the False Acceptance Rate (FAR) by 32% to 41% under both acoustic conditions. This suggests that relying solely on PTQ for model compression is insufficient for on-device constrained keyword spotting tasks.

To illustrate the impact of QAT, we incorporate Absolute Cosine Regularization (ACR) on model weights during both S3RL (pre-training) and KWS (fine-tuning), and utilize either moving average quantization (***MA***) or dynamic quantization (***Dyn***) for activations. When comparing rows (2)(4) and (3)(5), we observe that models using PTQ (2)(3) outperform those with ACR QAT when using the same activation quantization approach (either MA or ***Dyn***). This suggests that ACR might be overly strict and challenging to apply, potentially due to the normal distribution pattern of model weights.

We further investigate the impact of QAT solely on S3RL by comparing rows (4)(6). Notably, without QAT on S3RL but applying only on KWS (6), the model outperforms both the S3RL QAT model (4) and the PTQ model (2) by a significant margin. This performance drop with S3RL QAT may be attributed to the limited model capacity resulting from the use of ACR and MA during S3RL pre-training.

This discovery suggests that a full-precision S3RL pre-training combined with QAT downstreaming (6) can help narrow the performance gap between the complete full-precision model (1) and the PTQ-only model (2), while there is still room for further improvements

Table 1: The results of the Alexa keyword spotting. w32a32 denotes 32-bit full-precision models, $w8a8_{MA}$ is a quantized model with 8-bit weights and 8-bit activation using moving average quantization and $w8a8_{Dyn}$ is a quantized model with 8-bit weights and 8-bit activation using dynamic quantization.

	S3RL	KWS	Final Precision	Relative FAR	
				Normal	Playback
(1)	FP	FP	w32a32	1.0	1.0
(2)	FP	FP	$w8a8_{MA}$	1.41	1.33
(3)	FP	FP	$w8a8_{Dyn}$	1.39	1.32
(4)	ACR+MA	ACR+MA	$w8a8_{MA}$	1.52	1.54
(5)	ACR+Dyn	ACR+Dyn	$w8a8_{Dyn}$	1.86	1.75
(6)	FP	ACR+MA	$w8a8_{MA}$	1.15	1.13
(7)	ACR+MA	FP	w32a32	1.10	1.10
(8)	MA	MA	$w8a8_{MA}$	1.89	1.81
(9)	Dyn	Dyn	$w8a8_{Dyn}$	1.02	1.01

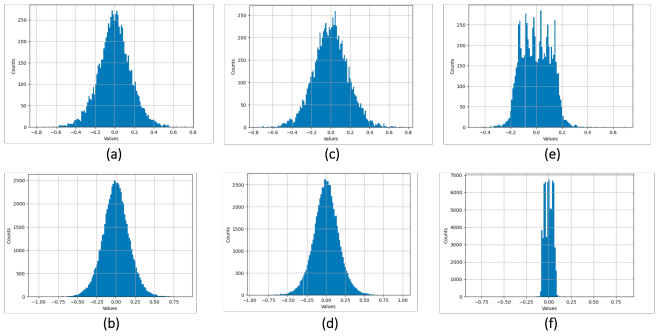

Fig. 2: These figures depict histograms of model weights from the first linear layer (a, c, e) and the attention output layer at the last transformer layer (b, d, f). From left to right, they show the full-precision model (1) (a,b), the model trained with ***Dyn*** (9) (c,d), and the model trained with ACR + ***Dyn*** (5) (e,f).

4.2. ACR vs Non-ACR

To further substantiate our previous assumption concerning ACR, we have generated histograms of model weights selected from two of the modules within the model. As depicted in Figure 2, we observe that the weights in the full-precision model already fall within the desired range of [-1, 1], and this consistency is maintained across different layers. In contrast, a comparison between full-precision model (a)(b) and ACR model (e)(f) illustrates that ACR results in a narrower weight range, with a tendency for the values to converge towards zero, especially in the deeper layer (f). Consequently, we believe that while ACR enables the selection of one of the quantized values, the normal distribution pattern of the weights [14] causes the majority of values to cluster around zero. Furthermore, since ACR does not perform forward-pass quantization, even slight perturbations near zero, which might be considered noise during post-quantization, actually contain valuable information for both self-supervised training and downstream tasks.

Based on previous results and observations, our approach involves solely quantizing activations during training, with model weights being quantized only after training. Rows (8) and (9) in Table 1 demonstrate the quantized model trained with ***Dyn*** (9) achieving comparable performance to the full-precision S3RL + KWS, surpassing the other w8a8 models. As illustrated in Figure 2, (c)(d) exhibit a weight distribution similar to that of the full-precision model (a)(b), which is more evenly distributed, enhancing the model's capacity and learning ability.

Table 2: Number of zero weights, utilization of quantized values and overall compression rate. All models are quantized to 8-bit (w8a8).

Setup	Training	Zeros ↓	Efficiency ↑	Compression ↑
KWS	FP	4.7%	41.8%	23.8%
	Dyn	3.7%	48.4%	24.1%
	ACR + Dyn	11.0%	20.0%	22.2%
S3RL + KWS	FP	4.3%	51.2%	23.9%
	Dyn	**3.7%**	**53.5%**	**24.1%**
	ACR + Dyn	11.0%	20.5%	22.3%

4.3. Efficiency Evaluation

To better illustrate the efficiency of QAT on S3RL and KWS, we introduce three metrics in Table 2: (1) number of zero weights, (2) utilization of quantized values and (3) overall compression rate. The utilization of quantized values is defined as "Efficiency" with:

$$e = (\sum_{i=1}^{q} \mathbf{1}_{s_i \in W^Q})/q \tag{6}$$

where $\{s_1, s_2, ..., s_q\}$ represents the set of quantized values, W^Q represents the quantized weights of a specific layer, and the final efficiency is averaged over all layers. By quantizing the full-precision weights (32-bit) into 8-bit, we achieve a compression rate of 25%. Furthermore, we were able to reduce the model size by pruning weights between $[-1/256, 1/256]$, which are quantized to 0. As presented in Table 2, the model trained with ***Dyn*** exhibits the lowest number of zeros in the final 8-bit model and also demonstrates the highest efficiency in utilizing quantized values. Additionally, we observed that the efficiency of using quantized values is higher with S3RL for all setups compared to the model without S3RL, indicating that S3RL enhances the efficiency of using quantized values.

5. CONCLUSION

Through our exploration of various Quantization-Aware Training (QAT) methods, we have determined that Absolute Cosine Regularization (ACR) is excessively restrictive for model weights in on-device modeling, primarily due to the normal distribution pattern of the weights. Instead, we have found that a combination of dynamic quantization on activations without ACR produces the best results, with performance approaching that of the full-precision model in an 8-bit model. Our analysis also indicates that incorporating Self-Supervised Speech Representation Learning (S3RL) and QAT can further enhance the efficiency of utilizing quantized values. In summary, our findings suggest that utilizing Non-ACR with Dynamic activation quantization for both self-supervised speech representation learning and keyword spotting fine-tuning can significantly reduce computational costs while compressing the model for on-device constraints without sacrificing performance. This approach renders the model more practical for real-world applications with limited resources.

6. REFERENCES

[1] Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass, "An Unsupervised Autoregressive Model for Speech Representation Learning," in *Interspeech*, 2019.

[2] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *NeurIPS*, 2020.

[3] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2021.

[4] Yu-An Chung, Yu Zhang, Wei Han, Chung-Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu, "w2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," in *ASRU*, 2021.

[5] Gene-Ping Yang, Sung-Lin Yeh, Yu-An Chung, James Glass, and Hao Tang, "Autoregressive predictive coding: A comprehensive study," *IEEE Journal of Selected Topics in Signal Processing*, 2022.

[6] Yuanchao Li, Yumnah Mohamied, Peter Bell, and Catherine Lai, "Exploration of a self-supervised speech model: A study on emotional corpora," in *SLT*, 2023.

[7] Holger Severin Bovbjerg and Zheng-Hua Tan, "Improving label-deficient keyword spotting using self-supervised pretraining," *arXiv preprint arXiv:2210.01703*, 2022.

[8] Chenyang Gao, Yue Gu, Francesco Caliva, and Yuzong Liu, "Self-supervised speech representation learning for keyword-spotting with light-weight transformers," in *ICASSP*, 2023.

[9] Gene-Ping Yang, Yue Gu, Qingming Tang, Dongsu Du, and Yuzong Liu, "On-Device Constrained Self-Supervised Speech Representation Learning for Keyword Spotting via Knowledge Distillation," in *Interspeech*, 2023.

[10] Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu, "TernaryBERT: Distillation-aware ultra-low bit BERT," in *EMNLP*, 2020.

[11] Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jin Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King, "BinaryBERT: Pushing the limit of BERT quantization," in *ACL*, 2021.

[12] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha, "Learned step size quantization," in *ICLR*, 2020.

[13] Hieu Duy Nguyen, Anastasios Alexandridis, and Athanasios Mouchtaris, "Quantization aware training with absolute-cosine regularization for automatic speech recognition," in *Interspeech*, 2020.

[14] Kai Zhen, Hieu Duy Nguyen, Raviteja Chinta, Nathan Susanj, Athanasios Mouchtaris, Tariq Afzal, and Ariya Rastrow, "Sub-8-Bit Quantization Aware Training for 8-Bit Neural Network Accelerator with On-Device Speech Recognition," in *Interspeech*, 2022.

[15] Antonio Polino, Razvan Pascanu, and Dan Alistarh, "Model compression via distillation and quantization," in *ICLR*, 2018.

[16] Jangho Kim, Simyung Chang, and Nojun Kwak, "PQK: Model Compression via Pruning, Quantization, and Knowledge Distillation," in *Interspeech*, 2021.

[17] Lu Zeng, Sree Hari Krishnan Parthasarathi, Yuzong Liu, Alex Escott, Santosh Cheekatmalla, Nikko Strom, and Shiv Vitaladevuni, "Sub 8-bit quantization of streaming keyword spotting models for embedded chipsets," in *TSD*, 2022.

[18] Yi Wei, Xinyu Pan, Hongwei Qin, Wanli Ouyang, and Junjie Yan, "Quantization mimic: Towards very tiny cnn for object detection," in *ECCV*, 2018.

[19] Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer, "Q-bert: Hessian based ultra low precision quantization of bert," in *AAAI*, 2020.

[20] Takanori Ashihara, Takafumi Moriya, Kohei Matsuura, and Tomohiro Tanaka, "Deep versus Wide: An Analysis of Student Architectures for Task-Agnostic Knowledge Distillation of Self-Supervised Speech Models," in *Interspeech*, 2022.

[21] Zheng Li, Zijian Wang, Ming Tan, Ramesh Nallapati, Parminder Bhatia, Andrew Arnold, Bing Xiang, and Dan Roth, "DQ-BART: Efficient sequence-to-sequence model via joint distillation and quantization," in *ACL*, 2022.

[22] Zechun Liu, Barlas Oguz, Aasish Pappu, Lin Xiao, Scott Yih, Meng Li, Raghuraman Krishnamoorthi, and Yashar Mehdad, "Bit: Robustly binarized multi-distilled transformer," in *NeurIPS*, 2022.

[23] Zilun Peng, Akshay Budhkar, Ilana Tuil, Jason Levy, Parinaz Sobhani, Raphael Cohen, and Jumana Nassour, "Shrinking bigfoot: Reducing wav2vec 2.0 footprint," in *sustainlp*. 2021, ACL.

[24] Ching-Feng Yeh, Wei-Ning Hsu, Paden Tomasello, and Abdelrahman Mohamed, "Efficient speech representation learning with low-bit quantization," *arXiv preprint arXiv:2301.00652*, 2022.

[25] Aaron van den Oord, Yazhe Li, and Oriol Vinyals, "Representation learning with contrastive predictive coding," *arXiv:1807.03748*, 2018.

[26] Dongwei Jiang, Wubo Li, Ruixiong Zhang, Miao Cao, Ne Luo, Yang Han, Wei Zou, Kun Han, and Xiangang Li, "A further study of unsupervised pretraining for transformer based speech recognition," in *ICASSP*, 2021.

[27] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou, "Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *CoRR*, 2016.

[28] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *CoRR*, 2016.

[29] Yoshua Bengio, Nicholas Léonard, and Aaron Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *CoRR*, 2013.