Low-bit quantization and quantization-aware training for small-footprint keyword spotting

Anonymous Author(s)
Affiliation
Address
email

Abstract

We investigate low-bit quantization to reduce computational cost of deep neural network (DNN) based keyword spotting (KWS). We propose approaches to further reduce quantization bits via integrating quantization into keyword spotting model training, which we refer to as quantization-aware training. Our experimental results on large dataset indicate that quantization-aware training can recover performance models quantized to lower bits representations. By combining quantization-aware training and weight matrix factorization, we are able to significantly reduce model size and computation for small-footprint keyword spotting, while maintaining performance.

Index Terms: keyword spotting, quantization-aware training, small-footprint.

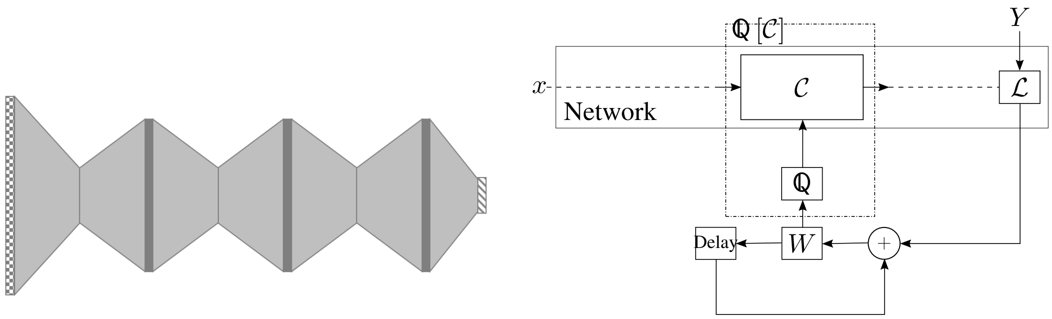
1 Introduction

Keyword spotting is the task of detecting particular words of interest in an audio stream. It has been an active research area in speech recognition and widely used in applications. With recent increase in the popularity of voice assistant systems, small-footprint keyword spotters (KWS) have been attracting much attention [1–3]. For example, Alexa on Amazon Tap requires the KWS to run continuously under tight CPU, memory, latency, and power constraints. The device only starts streaming audio to the cloud when the KWS detects the wake word. Such embedded KWS must have high recall to make devices easy to use, as well as low false accepts to mitigate privacy concerns.

One type of small-footprint KWS are systems based on a single DNN or convolutional neural network (CNN) [1, 4–6]. The keyword posterior calculated by such DNN or CNN are smoothed with a sliding window and the keyword detection event is triggered if the smoothed posterior exceeds a pre-defined threshold. The trade off between balancing false rejects and false accepts is performed by tuning a threshold. Context information is incorporated by stacking frames in the input. When deployed on device such KWS are always quantized. 16 bit and 8 bit quantizations are common in the industry [7–13]. Since the keyword models in such KWS are usually trained using full-precision arithmetic, quantization degrades their performance on device. One approach to mitigate that degradation is by using quantization-aware training. Quantization-aware training considers the quantized weights in full precision representation in order to inject the quantization error into training. This method enables the weights to be optimized against quantization errors.

In this work, we use quantization-aware training to build a very small-footprint low-power KWS. To train the wake word model, we employ quantization-aware training as a final training stage. We find that 8 bit and 4 bit quantized KWS models can be trained successfully by using quantization-aware training.

The paper is organized as follows: Section 2 introduces keyword spotting system and quantization-aware training approach. Experiments follow in Section 3. Conclusions are in Section 4.



(a) (b)

Figure 1: (a) Single-stage DNN KWS architecture used in this work. The DNN consists of 3 pairs of SVD-bottlenecks (light gray) followed by softmax (diagonal stripes). Each pair of SVD-bottleneck transformations is a sequence of 2 affine transforms with a simple linear activation between them, followed by a sigmoid activation layer (dark gray). (b) Scheme of quantization-aware training.

2 Keyword Spotting System and Quantization-Aware Training

Our keyword spotter is a single-stage DNN with 50k parameters [14]. The DNN operates on 20-dimensional log mel filter-bank energies (LFBE) acoustic features calculated over 25ms frames with a 10ms frame shift and stacked in 620-dimensional input windows, with 20 frames left and 10 frames right context. The DNN has 6 hidden layers with dimensions 39 and 128, shown in Figure 1a. The 128-unit layers are followed by sigmoid activation. There is simple linear activation after 39-unit layers. Such pairs of affine transformations represent an SVD approximation of one dense 128×128 layer [15, 16, 6]. The output layer in the DNN is softmax over 2 output states and represents the posterior distribution over the states $\{$'triphone $\in$ keyword', 'triphone $\notin$ keyword'$\}$. The DNN is trained using multi-target cross-entropy loss [14].

We use dynamic quantization approach, where shifts and scales for quantizing DNN weight matrices are calculated independenlty column-wise. This is similar to ”buketing” [17] or ”per-channel” [18] quantization with technical differences. Also, the inputs are quantized row-wise on the fly during the forward pass. This approach has better precision than static, single shift and scale quantization [11] (cf. TensorRT implemenation [19]). The software forward propagation implementation leverages hardware-specific SIMD operations to accelerate and parallelize quantized multiplications.

The accuracy loss due to quantization is incorporated via quantization-aware training, Figure 1b). We inject quantization errors at each DNN component C by enabling quantization of weights W and inputs during forward propagation. This transfers the quantization errors to the loss function $\mathcal{L}$ during back-propagation, calculated in full precision using quantized forward activation values $\mathbb{Q}[C]$. Quantization aware training is used as a final fine-stage tuning ensuring that the output of the final quantized model has matching accuracy with the full precision model. In contrast with [13], the model is not considered as floating point during the forward-pass, but the updates computed in true quantized form are passed onto floating point weights and then quantized. We do not use stochastic perturbation [12].

We test 16 bit, 8 bit, hybrid 4-8 bit, and 4 bit KWS quantization. For 16 bit, 8 bit and 4 bit quantizations, all layers are quantized using indicated bit-width. For 4-8 bit quantization, the first SVD-bottleneck pair and each consecutive 2nd bottleneck layer is quantized as 8 bit, and each 1st bottleneck layer is 4 bit. This is because the first hidden and each 2nd bottleneck-layer receive non-squeezed input with a potentially large dynamic range, and thus may require larger bit-precision for quantization. At the same time, the layers following sigmoid activation have inputs in the range $[0, 1]$, and therefore may allow lower bit quantization including 4 bits.

3 Experimental Results

The keyword 'Alexa' is chosen for our experiments. We use an in-house 500 hrs far-field corpus of diverse far-field speech data and a similar composition 100 hrs dataset for evaluation. We evaluate all models using end-to-end Detection Error Tradeoff (DET) curves, which describe the models' miss rate vs. false accept rate (FAR), as well as DET area under curve (AUC). For training, we use GPU-based distributed DNN training method described in [20]. The training is organized into 3 stages: In the 1st stage a small ASR DNN with 3 hidden layers of 128 units is pre-trained from random initialization and using full ASR phone-targets obtained from a large, production ASR system. In the 2nd stage, the KWS DNN is trained from the 1st-stage ASR DNN by adding keyword targets and performing multi-task training with the keyword and the ASR targets as regularization. In the 3rd stage, SVD bottlenecks are introduced and the model is multi-task trained with the SVD bottlenecks. The 1st-stage DNN is trained for a fixed duration of 12 epochs. The 2nd and 3rd stages are 20 epochs each. Exponential decaying learning rate is used with the initial value of 0.000125 and the decay factor of 2 for the first few epochs and 1.2 for remaining epochs. The final DNN is first 'naively' quantized using 16 bit, 8 bit, 4 bit, or hybrid 4-8 bit scheme and quantization-aware trained for another 20 epochs, using the same exponential learning rate decay schedule.

The performance of the 'naively quantized' models is shown in Table 1. We observe that 16 bit and 8 bit quantization show little degradation in KWS accuracy well under 1% AUC, while 4-8 bit hybrid and 4 bit quantization lead to significant performance degradation of 16.8% and 221.6% AUC, respectively. Therefore, we are interested in the effect of quantization-aware training on those latter two situations. Those results are shown in Table 2. In the hybrid 4-8 bit quantized model, quantization-aware training recovers close to 90% of the accuracy loss. After quantization-aware training, that model performs only slightly worse than the full-precision model at roughly 10% reduction in memory footprint compared to 8 bit-quantized version. The 4-bit quantized model shows yet greater performance gains due to quantization-aware training, from 2.2159 AUC to 1.41 AUC, or 36.4% reduction. A comparison of the end-to-end DET curves in Figure 2 shows that that model has significantly more reasonable DET (purple line), while the naively 4-bit quantized model is much worse (green line). Thus, if naively quantized 4-bit model showed 4x and 5x FAR of the full-precision model at same miss rate, the quantization-aware trained 4-bit model reduced that degradation to 40-50% in the range of interesting miss rates 5-15%. Such gains can be interesting for low-power applications, considering that the 4 bit-quantized model allows close to 40% additional reduction in memory footprint of the KWS.

Table 1: Relative AUC and model sizes of quantized KWS with respect to the baseline full-precision model. (Larger AUC is worse.)

	16 bit	**8 bit**	**4-8 bit**	**4 bit**
AUC	1.0003	1.0094	1.1684	2.2159
Model size	0.65	0.35	0.32	0.20

Table 2: AUC improvement of quantized models' performance using quantization-aware training.

	4-8 Bit	**4 Bit**
Quantized	1.1684	2.2159
QAT	1.0213	1.4103
Change	-12.6%	-36.4%

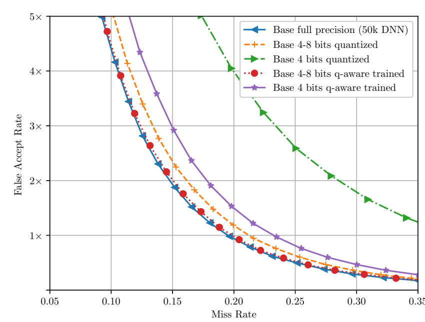


Figure 2: DET for full-precision, quantized, and quantization-aware trained 50k model. The DET curves for 16 bit and 8 bit quantized-models are not shown due to them not being significantly different from the full-precision model.

4 Conclusions

We present our work on applying quantization-aware training to reducing quantization bits of small-footprint keyword spotting models. Combined with weight matrix factorization, our method can significantly reduce the model size and computation cost of keyword spotting systems. Our experimental results indicate that with quantization-aware training 4-8 bit hybrid quantization maintains performance of full-precision model, while for 4-bit quantization performance gap can be significantly reduced.

References

[1] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4087–4091.

[2] M. Sun, V. Nagaraja, B. Hoffmeister, and S. Vitaladevuni, "Model shrinking for embedded keyword spotting." in *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*. IEEE, 2015, pp. 369–374.

[3] Q. He, G. W. Wornell, and W. Ma, "An adaptive multi-band system for low power voice command recognition," *Interspeech 2016*, pp. 1888–1892, 2016.

[4] P. Nakkiran, R. Alvarez, R. Prabhavalkar, and C. Parada, "Compressing deep neural networks using a rank-constrained topology." in *INTERSPEECH*, 2015, pp. 1473–1477.

[5] T. N. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting." in *INTERSPEECH*, 2015, pp. 1478–1482.

[6] G. Tucker, M. Wu, M. Sun, S. Panchapagesan, G. Fu, and S. Vitaladevuni, "Model compression applied to small-footprint keyword spotting," in *Proc. Interspeech*, 2016.

[7] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Advances in neural information processing systems*, 2015, pp. 3123–3131.

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

[9] L. Hou, Q. Yao, and J. T. Kwok, "Loss-aware binarization of deep networks," *arXiv preprint arXiv:1611.01600*, 2016.

[10] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained ternary quantization," *arXiv preprint arXiv:1612.01064*, 2016.

[11] C. Leng, H. Li, S. Zhu, and R. Jin, "Extremely low bit neural network: Squeeze the last bit out with ADMM," *arXiv preprint arXiv:1707.09870*, 2017.

[12] S. Wu, G. Li, F. Chen, and L. Shi, "Training and inference with integers in deep neural networks," *arXiv preprint arXiv:1802.04680*, 2018.

[13] D. Lee and B. Kim, "Retraining-based iterative weight quantization for deep neural networks," *arXiv preprint arXiv:1805.11233*, 2018.

[14] M. Sun, D. Snyder, Y. Gao, V. Nagaraja, M. Rodehorst, S. Panchapagesan, N. Ström, S. Matsoukas, and S. Vitaladevuni, "Compressed time delay neural network for small-footprint keyword spotting," in *Proc. Interspeech*, 2017, pp. 3607–3611.

[15] T. N. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, and B. Ramabhadran, "Low-rank matrix factorization for deep neural network training with high-dimensional output targets," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6655–6659.

[16] A. Senior and X. Lei, "Fine context, low-rank, softplus deep neural networks for mobile speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 7644–7648.

[17] A. Polino, R. Pascanu, and D. Alistarh, "Model compression via distillation and quantization," *arXiv preprint arXiv:1802.05668*, 2018.

[18] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," *arXiv preprint arXiv:1806.08342*, 2018.

[19] S. Migacz, “8-bit inference with TensorRT,” http://on-demand.gputechconf.com/gtc/2017/presentation/s7310-8-bit-inference-with-tensorrt.pdf, 2017.

[20] N. Strom, “Scalable distributed dnn training using commodity gpu cloud computing,” in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.