Simon Christian Klein*, Jonas Kantic, and Holger Blume

Fixed Point Analysis Workflow for efficient Design of Convolutional Neural Networks in Hearing Aids

Abstract: Neural networks (NN) are a powerful tool to tackle complex problems in hearing aid research, but their use on hearing aid hardware is currently limited by memory and processing power. To enable the training with these constrains, a fixed point analysis and a memory friendly power of two quantization (replacing multiplications with shift operations) scheme has been implemented extending TensorFlow, a standard framework for training neural networks, and the Qkeras package [1, 2].
The implemented fixed point analysis detects quantization issues like overflows, underflows, precision problems and zero gradients. The analysis is done for each layer in every epoch for weights, biases and activations respectively. With this information the quantization can be optimized, e.g. by modifying the bit width, number of integer bits or the quantization scheme to a power of two quantization. To demonstrate the applicability of this method a case study has been conducted. Therefore a CNN has been trained to predict the Ideal Ratio Mask (IRM) for noise reduction in audio signals. The dataset consists of speech samples from the TIMIT dataset mixed with noise from the Urban Sound 8k- and VAD-dataset at 0 dB SNR. The CNN was trained in floating point, fixed point and a power of two quantization. The CNN architecture consists of six convolutional layers followed by three dense layers.
From initially 1.9 MB memory footprint for 468k float32 weights, the power of two quantized network is reduced to 236 kB, while the Short Term Objective Intelligibility (STOI) Improvement drops only from 0.074 to 0.067.
Despite the quantization only a minimal drop in performance was observed, while saving up to 87.5 % of memory, thus being suited for employment in a hearing aid.

Keywords: convolutional neural networks, quantization, hearing aids, speech enhancement, audio signal processing.

https://doi.org/10.1515/cdbme-2021-2201

***Simon Christian Klein:** Institute for Microelectronic Systems, Appelstraße 4, Hanover, Germany, e-mail: simon.klein@ims.uni-hannover.de
Jonas Kantic, Holger Blume: Institute for Microelectronic Systems, Hanover, Germany

1 Introduction

In recent years deep neural networks (DNN) have become an important tool that can also be used for audio applications. Tasks such as noise reduction, speaker separation or speech recognition can be solved with ever increasing accuracy. In order to benefit from these advances in the hearing aid domain, the limitations of hearing aid processors have to be taken into consideration during design and training of a DNN. A good way of achieving this goal is the use of quantization, which reduces the memory footprint of the DNN and speeds up hardware operations.
Current frameworks, such as TensorFlow Lite, support quantized training which is, however, limited to server architectures, such as the TPU [10]. The quantization scheme of the TPU architecture is fixed, thus no further application specific optimization can be applied, like for example adjusting the bit width or exploiting the benefits of different quantization schemes, e.g. power of two quantization. This work makes the following contributions to this end:

- Implementation of a fixed point analysis within the Qkeras framework
- Extension of the Qkeras framework with a power of two quantization
- validation of the method by training a neural network for a noise reduction application.

1.1 Related Work

Fixed point arithmetic can significantly reduce computational power, memory and energy requirements and is thus a common approach in low power and also in hearing aid applications [3, 4]. As is has been shown, these advantages can also be applied to neural networks, including the use of a power of two quantization [5]. When a value is multiplied by a power of two, the multiplication can be represented as a

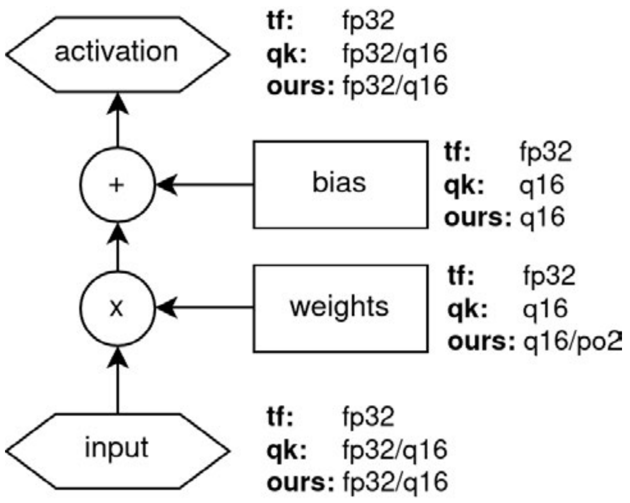

Figure 1: Quantization diagram of a feed forward layer. tf = TensorFlow layer, qk = Qkeras layer, ours = extended Qkeras layer, fp32 = floating point 32 bit, q16 = fixed point 16 bit, po2 = power of two quantization

shift operation in hardware. This speeds up the inference, but also reduces the memory requirement significantly, because only the power of two needs to be stored (e.g. instead of saving the value $2^7 = 128$ with 8 Bit, only 3 Bits are needed to store the power 7). The training of quantized networks is supported by frameworks such as TensorFlow Lite and Qkeras to different degrees, whereby TensorFlow Lite is more restricted in terms of quantization schemes and extensibility. Up to now Qkeras supports only fixed point quantization with arbitrary bit with.

In the use case of noise reduction, fully connected neural networks (FCNN) [6], convolutional neural networks (CNNs) [7], and recurrent neural networks (RNNs) [8] have been successfully trained so far. Real-time capability and limited memory in the range of less than 100 k B were partially considered [7, 8], but floating point arithmetic was used instead of fixed point arithmetic, which is most certainly not available on a hearing aid processor. In the following chapter, first the implementation of the fixed point analysis is described followed by an evaluation using the case study of a C NN trained for noise reduction.

2 Algorithmic Framework

During the training of feedforward NNs, there are several elements which can be quantized independently of each other. The corresponding dataflow is depicted in figure 1: first an input tensor is multiplied by weights, then a bias is added to it. Afterwards, an activation is applied to the tensor, which is then feed as an input of the next layer. In the TensorFlow training routine, input, weights, bias and output are available as float32 data type. In Qkeras the inference is altered in order to represent quantization. The parameters (weights and bias) are

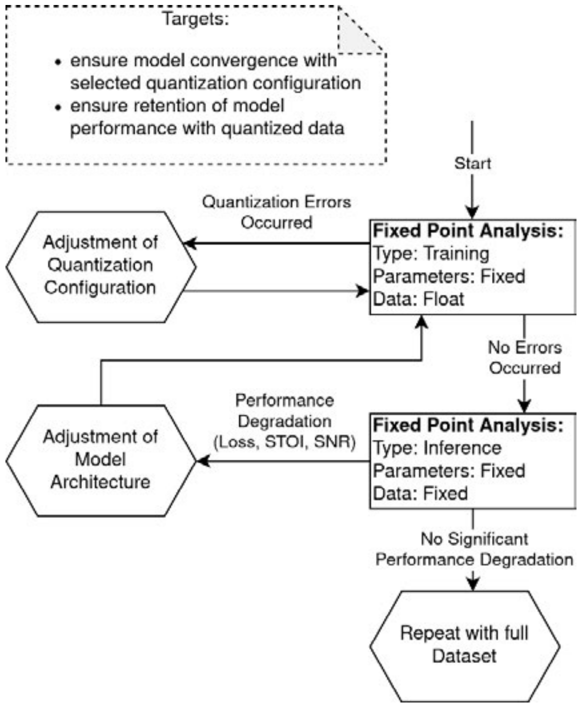
Targets:
• ensure model convergence with selected quantization configuration
• ensure retention of model performance with quantized data
Start
Quantization Errors Occurred
Adjustment of Quantization Configuration
Fixed Point Analysis:
Type: Training
Parameters: Fixed
Data: Float
No Errors Occurred
Performance Degradation (Loss, STOI, SNR)
Adjustment of Model Architecture
Fixed Point Analysis:
Type: Inference
Parameters: Fixed
Data: Fixed
No Significant Performance Degradation
Repeat with full Dataset

Figure 2: Workflow of fixed point analysis with representative subset of full dataset

quantized for each forward pass, while the parameters are still saved in floating point format. A power of two quantization was additionally implemented. Because the bias is more sensitive to quantization, is was only applied on the weights.

If the parameters were not additionally stored in floating point format, small gradients during backpropagation or small adjustments of the weights could not be displayed and the model would not converge during training.

In Contrast to the parameters the activations are not stored in floating point format during quantization with Qkeras.

Instead, quantized activation functions are used for gradient calculation. This results in the fact that differences from one epoch to the next, which are below the quantization resolution, results in the gradient being zero. Thus, the model would not converge.

Therefore, in this work a two-step training process according to figure 2 is proposed, where the network architecture is first trained with the partially (parameters only) quantized Qkeras layers. During training, quantization errors, as well as overflows and underflows for each parameter (weights, bias), as well as activations, are stored and displayed to the developer. Thus, the influence of the quantization can be evaluated and the selected quantization can be adjusted accordingly. In order to iterate in a reasonable timeframe over different fixed point formats, a representative subset of the complete data set should be used for the fixed point analysis.

If the quantization errors, overflows and underflows are within acceptable limits, the trained weights are transferred to a fully quantized version, including quantization of activations, and the network is retrained. When quantizing the activations in

addition to weights and bias, the effects of inadequate quantization becomes visible, therefore quantization problems are also examined in this step of the fixed point analysis and the model is evaluated with respect to the performance.
If the performance is insufficient, the network architecture should be adjusted and the network should be trained partially quantized again. If the performance is sufficient even with a fully quantized network, the full data set can now be used for training.

3 Evaluation

In this section, the presented fixed point analysis and the implemented power of two quantization are evaluated for the training of a CNN with the goal of noise reduction application. First, the training process of an architecture with a fixed point quantization of 3.13, i.e. 1 sign bit, 2 integer bits and 13 fractional bits, is examined and the evolution of the quantization errors is shown. Subsequently, the same architecture is trained with a power of two quantization. Only the weights are quantized in this way, the bias and the activations are still in the 3.13 fixed point format.
The input of the neural network is the Mel frequency spectrum with 64 bins, calculated based on the current and 15 past audio frames. The Mel spectrum is computed frame by frame, where one frame consists of 256 audio samples at a sampling frequency of 16 kHz, the frames overlap by 128 samples. This results in an input spectrum with the dimensions 16 * 64.As output of the CNN the Ideal Ratio Mask [9] for the next frame is calculated, thus estimating the ratio between noise and speech for each frequency bin for the current frame. This setup is real time capable, assuming, the inference and the Mel frequency spectrum can be calculated in 8 ms. For the training clean speech signals from the TIMIT data set were mixed with noise from the Urban Sound 8k- and the VAD- data set at 0 dB SNR level.
The CNN itself is constructed as shown in table 1. Every layer is followed by a relu activation function, expect for the last layer, which uses a sigmoid activation as a regression output. All convolutional and separable convolutional layers work with 3 x 3 kernels. Using floating point weights, the baseline network has a memory footprint of 1.9 MB.

Table 1: CNN architecture for the case study. To compute the memory footprint in bytes, the parameter number needs to be multiplied by the size of the used datatype (e.g. times 4 for fp32). The total number of parameters is 468k, thus for fp32 the memory footprint is 1.9 MB.

Layers	Filters/Neurons	Parameters Number
Input	-	-
Conv2D	8	80
SeparableConv2D	16	216
SeparableConv2D	16	416
AveragePooling2D with 1 x 2 kernel	-	
SeparableConv2D	32	688
AveragePooling2D with 1 x 2 kernel	-	
SeparableConv2D with 2 x 2 Stride	64	2400
SeparableConv2D	64	4736
AveragePooling2D with 4 x 4 kernel	-	
Fully Connected	512	131584
Fully Connected	512	262656
Fully Connected	64	66177

4 Fixed Point Analysis

The results of the fixed point analysis can be divided into two categories: Boundary violations and quantization violations. Boundary violations indicate the lack of integer bits, while quantization violations indicate the lack of fractional bits. To exemplify the occurrence and consequences, two CNNs, one with a 3.13 fixed point quantization and one with a power of two quantization were trained on a representative subset of the full data set.
No boundary violation of the parameters could be detected for any of the networks. The maximum value of the activation was with a value of nearly 300 larger than 4, which would have been still representable with 2 bits. However only approximately 2% of the activations lie over the maximum representable value. Due to the training with the representative subset, the performance improves only by 0.02 in terms of the STOI, but is expected to be higher with the full dataset. A post-training with additionally quantized activation, as provided in the presented fixed point analysis, achieves a similar performance on a validation data set.
Similar to the boundary violations, the smallest occurring value of 10e-8 also falls far short of the quantization resolution

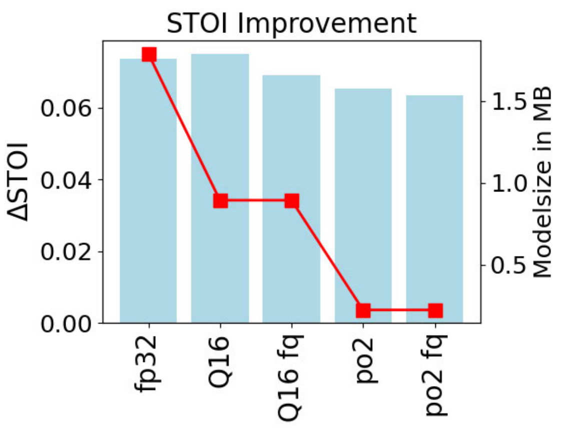

Figure 3: STOI Improvement (bar chart) and model size (red line) depending on the quantization. fp32 = floating point model; Q16 = fixed point model; po2 = power of two quantization model; fq = fully quantized

of 0.0001 for the quantization violation. In order to determine the percentage of quantization violations in a meaningful way, all quantized values that differ more than 5% from the actual floating point value are evaluated as quantization violations. This shows a clear difference between the 3.13 quantization, which has only 3 % quantization errors related to the parameters, in contrast to the power of two quantization, where the quantization errors account for up to 80 %. This is due to the wide dispersion of the weights, but a determination of the performance shows that the STOI of both quantization types differs only insignificantly.

Based on the fixed point analysis three CNNs were trained: without quantization, with 3.13 fixed point format and a power of two quantization. The activations of fixed point model and the power of two model were quantized and validated again after retraining, see figure 3. Comparing to the floating point reference with a STOI of 0.074 to the fully quantized power of two model, the STOI drops by only 15 %. But with only 236 kB against 1.9 MB of memory footprint nearly 87.5 % of the memory can be saved. While the achieved performance is comparable to previously published results, the proposed fixed point method of this work allows the training of fully quantized CNNs employable on hearing aid processors.

5 Conclusion

In this work we proposed a fixed point analysis extending the commonly used TensorFlow and Qkeras framework. Furthermore, we implemented a p ower of two quantization scheme to aid the development of extremely efficient CNNs for hearing aid devices. The presented methods were evaluated on CNNs for noise reduction, showing only a small drop in performance with quantization, but enabling significant memory savings.

Author Statement

This work was supported by the DFG Cluster of Excellence EXC 1077/1 “Hearing4all”.

References

[1] Abadi, Martín et al., 2016. Tensorflow: A system for large-scale machine learning. In 12th Symposium on Operating Systems Design and Implementation. pp. 265–283.

[2] Claudionor N. Coelho Jr. et al., 2020 Automatic deep heterogeneous quantization of Deep Neural Networks for ultra low-area, low-latency inference on the edge at particle colliders.

[3] Lee, Yu-Chi, Tai-Shih Chi, and Chia-Hsiang Yang. 2020. “A 2.17-MW Acoustic DSP Processor With CNN-FFT Accelerators for Intelligent Hearing Assistive Devices.” IEEE Journal of Solid-State Circuits 55 (8): 2247–58.

[4] Gerlach, Lukas, Guillermo Paya-Vaya, and Holger Blume. 2019. “KAVUAKA: A Low Power Application Specific Hearing Aid Processor.” In 2019 IFIP/IEEE 27th International Conference on Very Large Scale Integration (VLSI-SoC), 99–104. Cuzco, Peru: IEEE.

[5] Nayak, Prateeth, David Zhang, and Sek Chai. 2019. “Bit Efficient Quantization for Deep Neural Networks.” ArXiv:1910.04877 [Cs], October.

[6] Aubreville, Marc, Kai Ehrensperger, Tobias Rosenkranz, Benjamin Graf, Henning Puder, and Andreas Maier. 2018. “Deep Denoising for Hearing Aid Applications.” 2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC), September, 361–65

[7] Chakrabarty, Soumitro, DeLiang Wang, and Emanuel A. P. Habets. 2018. “Time-Frequency Masking Based Online Speech Enhancement with Multi-Channel Data Using Convolutional Neural Networks.” In 2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC), 476–80. Tokyo: IEEE.

[8] Schröter, Hendrik, Tobias Rosenkranz, Alberto N. Escalante-B., Pascal Zobel, and Andreas Maier. 2020. “Lightweight Online Noise Reduction on Embedded Devices Using Hierarchical Recurrent Neural Networks.” ArXiv:2006.13067 [Cs, Eess], June.

[9] Yuxuan Wang, Arun Narayanan, and DeLiang Wang. 2014. “On Training Targets for Supervised Speech Separation.” IEEE/ACM Transactions on Audio, Speech, and Language Processing 22 (12): 1849–5

[10] Jouppi, Norman, Cliff Young, Nishant Patil, and David Patterson. 2018. “Motivation for and Evaluation of the First Tensor Processing Unit.” IEEE Micro 38 (3): 10–19.