Practical Approaches to DNN Quantization

Dwith Chenna

Senior Embedded DSP Eng., Computer Vision

Magic Leap Inc.

Contents

Why Quantization?

- Quantization is a powerful tool to enable deep learning on edge devices
- Resource constrained hardware with limited memory and low power requirement

Why Quantization?

- Model compression: Up to 4x smaller (float32 to int8) network size and memory bandwidth

- Latency reduction: Up to 2x-3x times, int8 compute is significantly faster compared to float32 [1]

- Trade-off: Potential effects on the model accuracy

Quantization Scheme

- Convert full precision float-point numbers to int8 [2]

$$q = round\left(\frac{r}{s} + z\right)$$

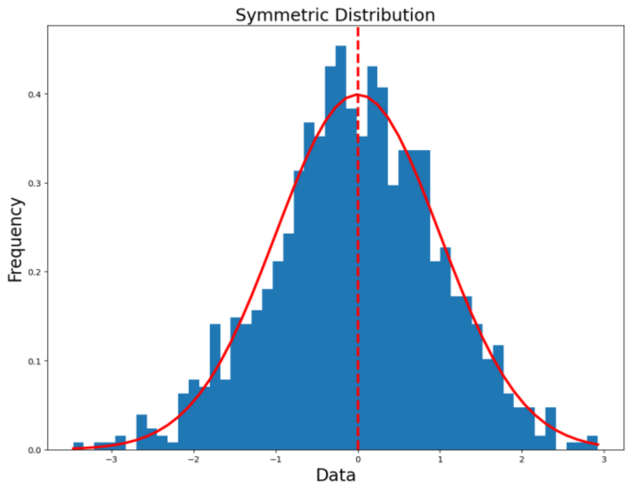
q - quantized value, r - real value, s - scale, z - zero point

- Quantized value to float-point representation $r = s\left(q - z\right)$
- In case of float-point distribution, we obtain scale and zero point as:

$$s = \frac{(r\max - r\min)}{(q\max - q\min)} \qquad z = round\left(q\max - \frac{r\max}{s}\right)$$

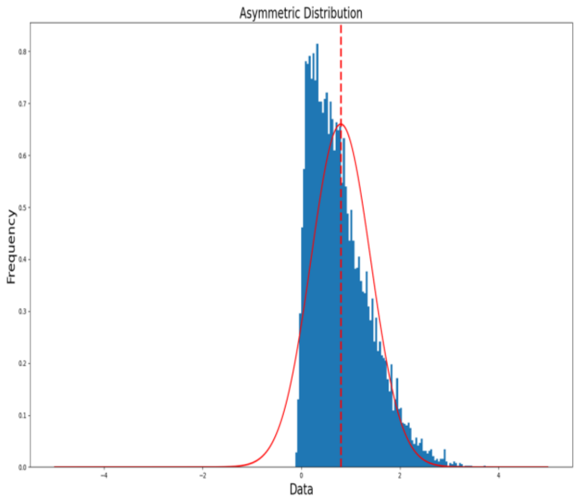
Quantization Scheme: Symmetric

- Assumes symmetric distribution for simplicity, zero point = 0
- Symmetric per tensor
  - Calculate scale for the entire tensor
- Symmetric per channel
  - Calculate scale for each channel of the tensor
- Computationally efficient



Quantization Scheme: Asymmetric

- Accounts for shifts in the distribution, better utilization of quantization range
- Asymmetric per tensor
  - Scale and zero point for the entire tensor
- Symmetric per channel
  - Scale and zero points for each channel of the tensor
- Better handling of diverse distributions



Contents

- Why Quantization?
- Quantization Scheme
- **Types of Quantization**
- Post Training Quantization
- Quantization Tools
- Network Architecture
- Calibration Dataset
- Min/Max Tuning
- Quantization Evaluation
- Quantization Analysis
- Quantization Aware Training
- Best Practices

Types of Quantization

- Post Training Quantization (PTQ)
  - Simple yet efficient
  - Uses already trained model and calibration dataset
- Quantization Aware Training (QAT)
  - Emulates inference-time quantization
  - Resource intensive as it needs retraining

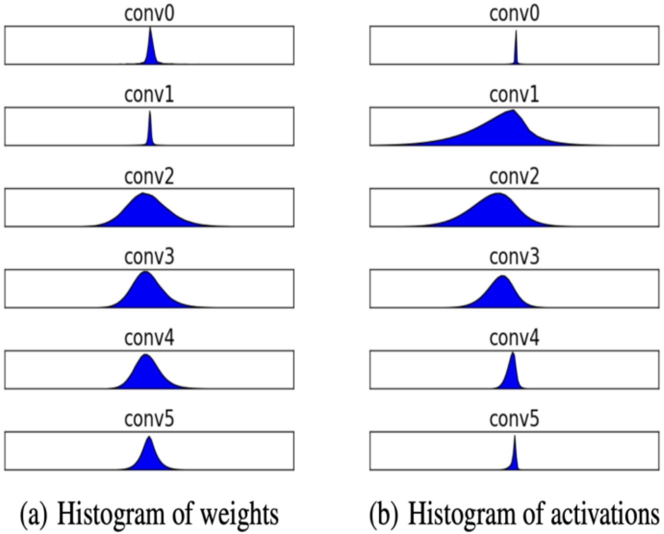
Post Training Quantization

- Dynamic Quantization
  - Weights are quantized ahead of time
  - Activations are quantized during inference (dynamic)

- Static Quantization
  - Weights and activations are quantized
  - Memory bandwidth and compute savings
  - Needs representative dataset

Post Training Quantization

- Best quantization scheme for deep neural networks?
- Weights: Symmetric per channel
  - Static distribution makes it easy for quantization
  - Weight distributions tend to be symmetric [3]
  - Symmetric per channel handles diversity in weight distribution



[Empirical distribution in a pre-trained network]

Post Training Quantization

- Activations: Asymmetric/Symmetric per tensor
  - Dynamic distribution per inference makes it difficult to find statistics
  - Approximation through representative/calibration dataset
  - Batch normalization enables better distributions for quantization

Contents

Quantization Tools: Tflite

- Tflite supports 8-bit integer PTQ [1]
- Quantization scheme
  - Weights: Symmetric per channel
  - Activations: Asymmetric per tensor
- Quantization analysis
  - Selective quantization with mixed precision (float32/16 + int8/int16)
  - Layerwise quantization error with custom metrics

Quantization Tools: Pytorch

- Pytorch supports 8-bit integer PTQ [4]
- Quantization scheme
  - Weights: (A)symmetric per tensor/channel
  - Activations: (A)symmetric per tensor/channel
- Quantization analysis
  - Layerwise quantization error through custom metrics

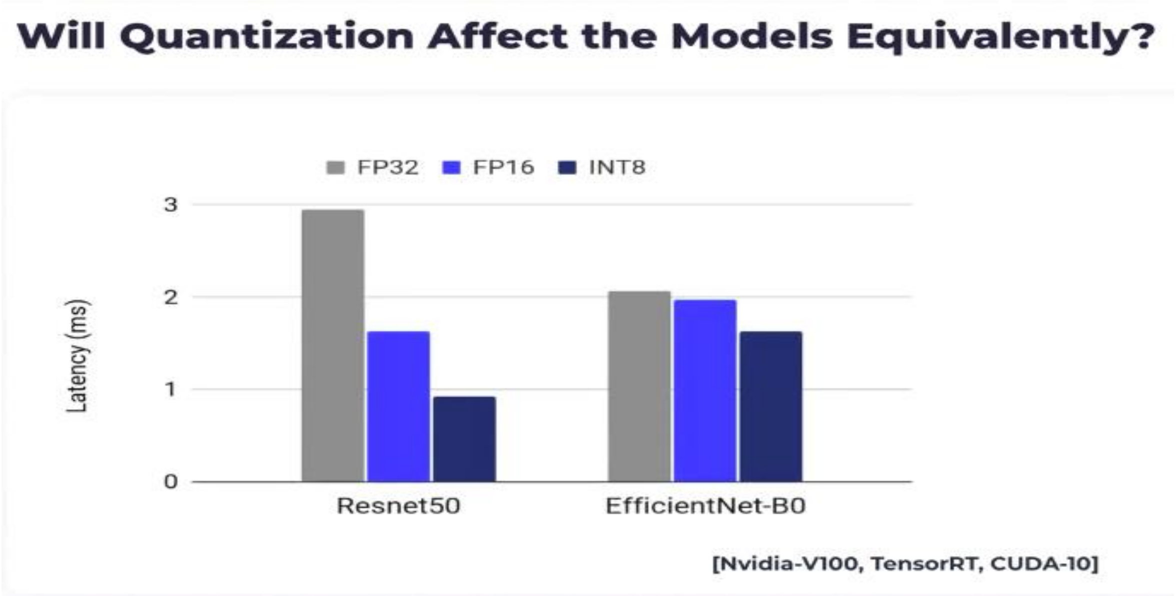
Contents

Network Architecture

- FLOPS is not everything!
- Network Architecture Search (NAS)
  - Most NAS based models (e.g., efficientNet) try to minimize compute
  - Results in deeper and leaner network that works well with cache-based systems

Network Architecture

- Efficient architecture for quantization [5]



Network Architecture

- Quantization aware
  - Larger models have redundancy which enables robustness to quantization
- Quantization scheme
  - Enable utilization of simpler and efficient quantization schemes

Network Architecture

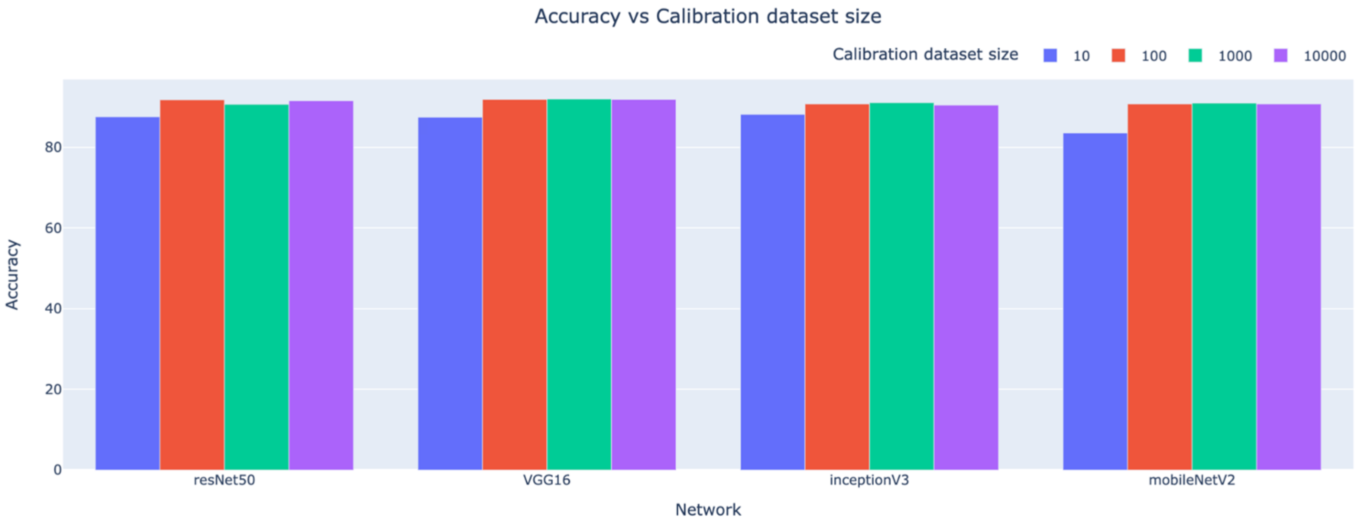
- Optimization tool chain
  - Aggressive layer fusion for optimal memory bandwidth
  - Optimal quantization parameter selection

- Hardware
  - Better suited for the hardware CPU/GPU/DSP/accelerator

Contents

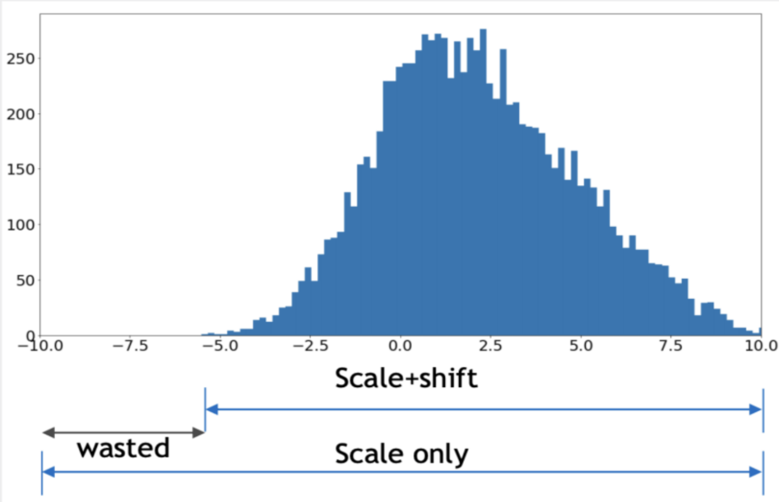
Calibration Dataset

- Representative dataset to estimate activation distribution
- Need to address diversity of the use case
- Size: ~100-1000 images are statistically significant [6]



Min/Max Tuning

- Minimize quantization error and eliminate outliers
- Trade-offs: range vs quantization error
- Mean/Standard deviation
  - Assuming normal distribution
  - Min/Max: mean +/- 3*STD



Min/Max Tuning

- Histogram
  - Ignore the last x% percent

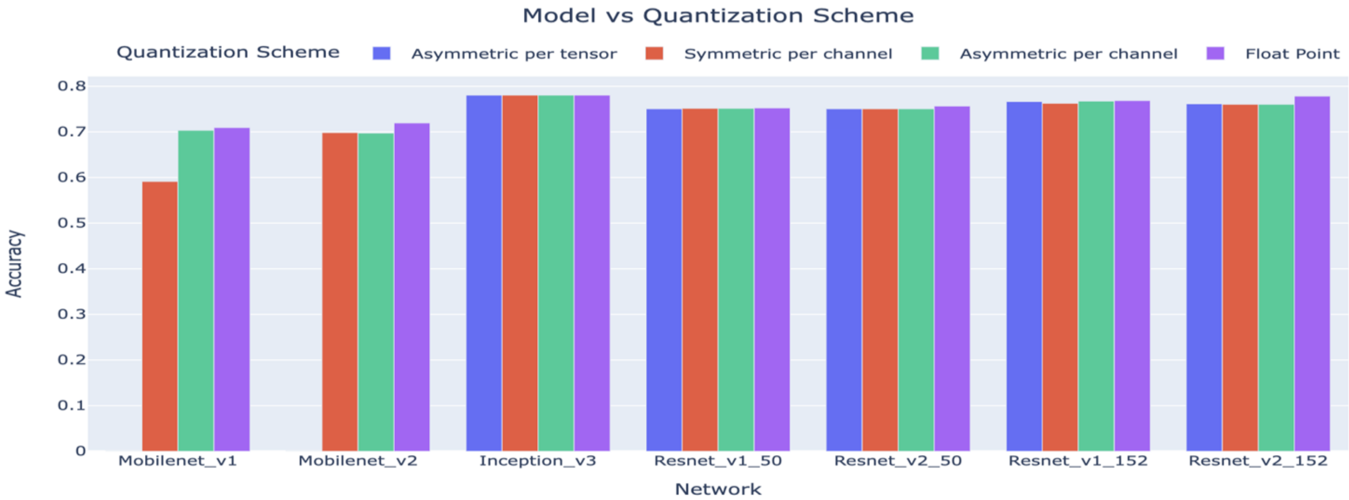
- Moving average (TensorFlow default)

- Search max/min (Pytorch/TensorRT)
  - Find histogram to cover most entropy

Contents

Quantization Evaluation

- Evaluate best fit quantization schemes to the model [2]
  - ResNet50: Symmetric per tensor
  - MobileNet: Asymmetric per channel



Quantization Evaluation

- Effects of quantization scheme on model accuracy [2]
- Classification accuracy of the quantized model

Network	Asymmetric, per-layer	Symmetric , per-channel	Asymmetric, per-channel	Activation Only	Floating Point
Mobilenet-v1_1_224	0.001	0.591	0.703	0.708	0.709
Mobilenet-v2_1_224	0.001	0.698	0.697	0.7	0.719
Nasnet-Mobile	0.722	0.721	0.74	0.74	0.74
Mobilenet-v2_1.4_224	0.004	0.74	0.74	0.742	0.749
Inception-v3	0.78	0.78	0.78	0.78	0.78
Resnet-v1_50	0.75	0.751	0.751	0.751	0.752
Resnet-v2_50	0.75	0.75	0.75	0.75	0.756
Resnet-v1_152	0.766	0.762	0.767	0.761	0.768
Resnet-v2_152	0.761	0.76	0.76	0.76	0.778

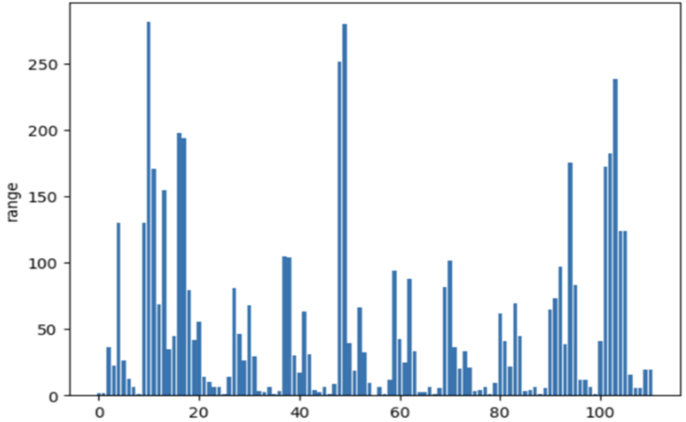
Contents

Quantization Analysis

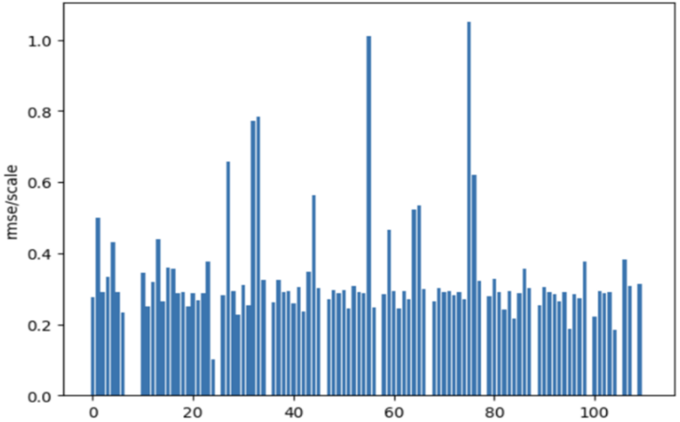
- What to do when quantization fails?
- Individual layer support for quantization
- Identifying few problematic layers will significantly improve performance
- Common pitfalls
  - Handling input/output quantization
  - Layer fusion before quantization

Quantization Analysis

- Analyse individual layers sensitivity to quantization [1]
- Selective quantization: mixed precision inference for testing



layer number (x-axis) vs activation range (y-axis)



root mean square error (rmse) vs activation range

There are many layers with wide ranges, and some layers have high rmse/scale values

Quantization Analysis

- Non-Linear activations: precision requirement and quantization support
  - ReLU/ReLU6 preferred over Sigmoid/LeakyReLU
- Weight/activation distribution: visualization or metrics for data distribution, i.e., range
- Layer fusion Conv + BN + ReLU / Conv + BN / Conv + ReLU before quantization

Quantization Analysis

- Use larger bit width for more sensitive layers, i.e., fully connected, network head
    - Int16 activation support in tflite

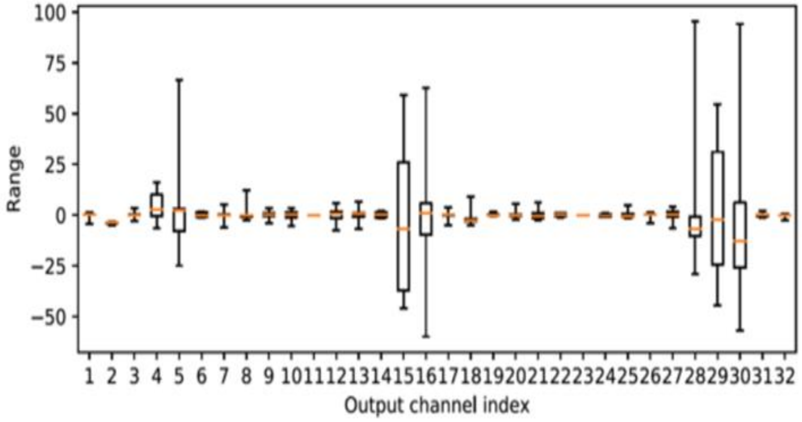
- Min/Max tuning: Outlier weights that cause all other weights to be less precise

Quantization Analysis

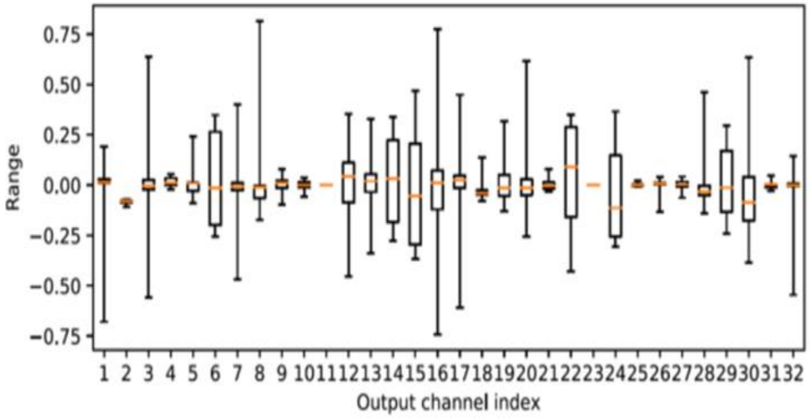
- Large difference in weight values for different output channels: more quantization error
  - Asymmetric/Symmetric per channel quantization
  - Weight equalization techniques to minimize the variation [7]

Quantization Analysis

- Weight equalization makes model quantization friendly



Pre-equalization box chart



Post-equalization box chart

Contents

Quantization Aware Training

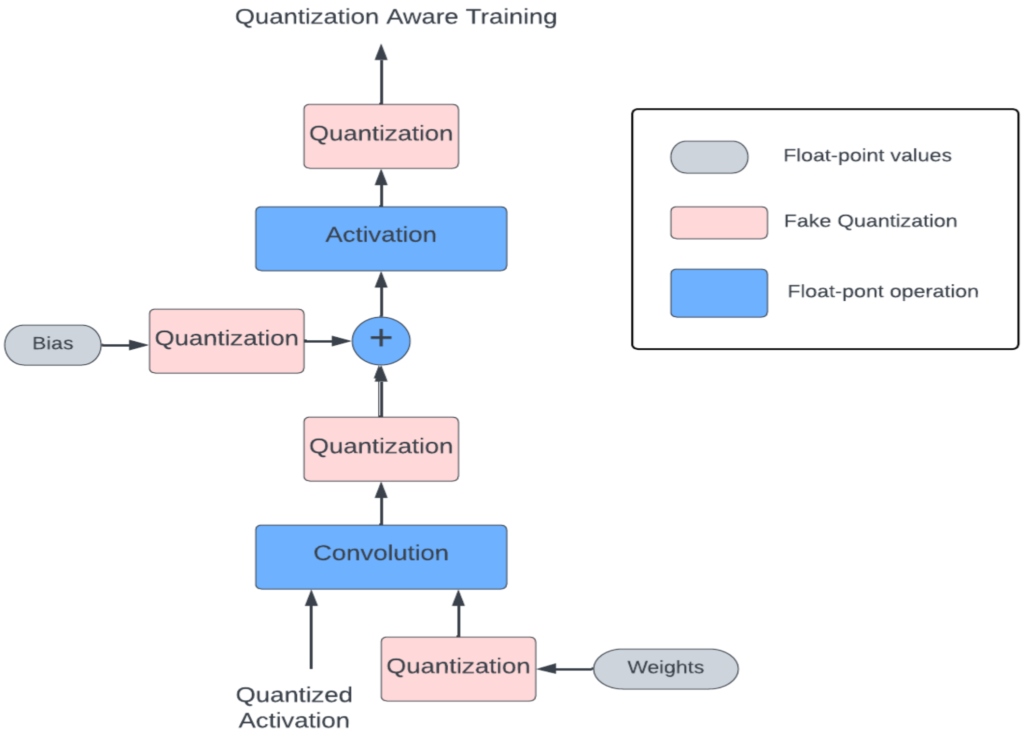
- When everything else fails!

- QAT is a fine-tuning process

- Start with trained floating-point model: with reduced momentum and learning rate

Quantization Aware Training

- Inserting quantization nodes during training [8]
- Simulate quantization using float-point operations
- Tune quantization parameters during training



Quantization Aware Training

- QAT is able to achieve < 1% accuracy degradation w.r.t to floating-point inference [2]

Model	fp32 Accuracy	PTQ best Calibration	PTQ best Accuracy	PTQ best Relative	QAT Accuracy	QAT Relative
MobileNet v1	71.88	99.9%	70.39	-2.07%	72.07	0.26%
MobileNet v2	71.88	99.99%	71.14	-1.03%	71.56	-0.45%
ResNet50 v1.5	76.16	Entropy	76.05	-0.14%	76.85	0.91%
ResNet152 v1.5	78.32	Entropy	78.21	-0.14%	78.61	0.37%
Inception v3	77.34	Entropy	77.54	0.26%	78.43	1.41%
Inception v4	79.71	99.99%	79.63	-0.10%	80.14	0.54%
ResNeXt50	77.61	Entropy	77.46	-0.19%	77.67	0.08%
ResNeXt101	79.30	99.999%	79.17	-0.16%	79.01	-0.37%
EfficientNet b0	76.85	Entropy	72.06	-6.23%	76.95	0.13%
EfficientNet b3	81.61	99.99%	80.28	-1.63%	81.07	-0.66%
Faster R-CNN	36.95	Entropy	36.82	-0.35%	36.76	-0.51%
Mask R-CNN	37.89	99.9999%	37.80	-0.24%	37.75	-0.37%
Retinanet	39.30	99.999%	39.19	-0.28%	39.25	-0.13%
FCN	63.70	Entropy	64.00	0.47%	64.10	0.63%
DeepLabV3	67.40	99.999%	67.50	0.15%	67.50	0.15%
GNMT	24.27	Entropy	24.53	1.07%	24.38	0.45%
Transformer	28.27	99.99%	27.71	-1.98%	28.21	-0.21%
Jasper	96.09	Entropy	96.11	0.02%	96.10	0.01%
BERT Large	91.01	99.999%	90.20	-0.89%	90.67	-0.37%

Table 7: Summary of Post Training Quantization and Quantization Aware Training. PTQ best reports the best accuracy and corresponding calibration for each model. QAT reports accuracy after fine-tuning starting from the best PTQ model.

Contents

Best Practices

- Model selection
  - NAS: Efficient architecture for quantization
- Quantization tools
  - Support for quantization schemes and analysis tools
- Calibration dataset
  - Representative dataset with ~100-1000 samples

Best Practices

- Quantization accuracy
  - Evaluate best-fit quantization scheme for the model
- Quantization analysis
  - Identify potentially problematic layers
- Quantization aware training
  - Fine tune model for quantization

References

1. https://www.tensorflow.org/lite/performance/post_training_quantization
2. Quantizing deep convolutional networks for efficient inference: A whitepaper [link]
3. Fixed Point Quantization of Deep Convolutional Networks [link]
4. https://pytorch.org/docs/stable/quantization.html
5. https://deci.ai/resources/achieve-fp32-accuracy-int8-inference-speed/
6. SelectQ: Calibration Data Selection for Post-Training Quantization[link]
7. AI Model Efficiency Toolkit (AIMET) [link]
8. Aspects and best practices of quantization aware training for custom network accelerators [link]