TensorQuant - A Simulation Toolbox for Deep Neural Network Quantization

Dominik Marek Loroch

MLHPC 13.11.2017

Fraunhofer ITWM

TensorQuant

- Quantization in Deep Learning
- The TensorQuant Toolbox
- Fixed-Point Experiments
- Plans for TensorQuant

Quantization in Deep Learning

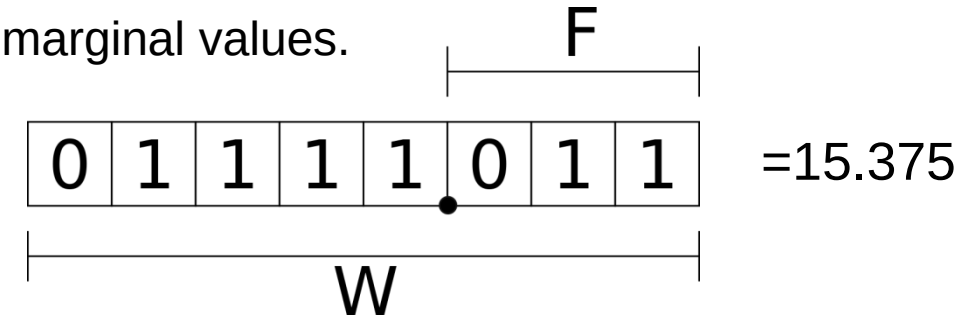
Quantization: Constraining a large set of values to a smaller, discrete one.

Quantization Methods:

- Clustering
- Binary / Ternary
- Logarithmic
- Fixed-Point Format
- Hashing
- ...

Fixed-Point Format:

- Number is stored like an integer (two's complement) with *W* bits word width.
- Last *F* bits are interpreted as the fractional part, i.e. the resolution is 2^{-F}
- Prevent overflow by saturating to the marginal values.



Fraunhofer ITWM

Quantization in Deep Learning

Table 1: The compression pipeline can save 35× to 49× parameter storage with no loss of accuracy.

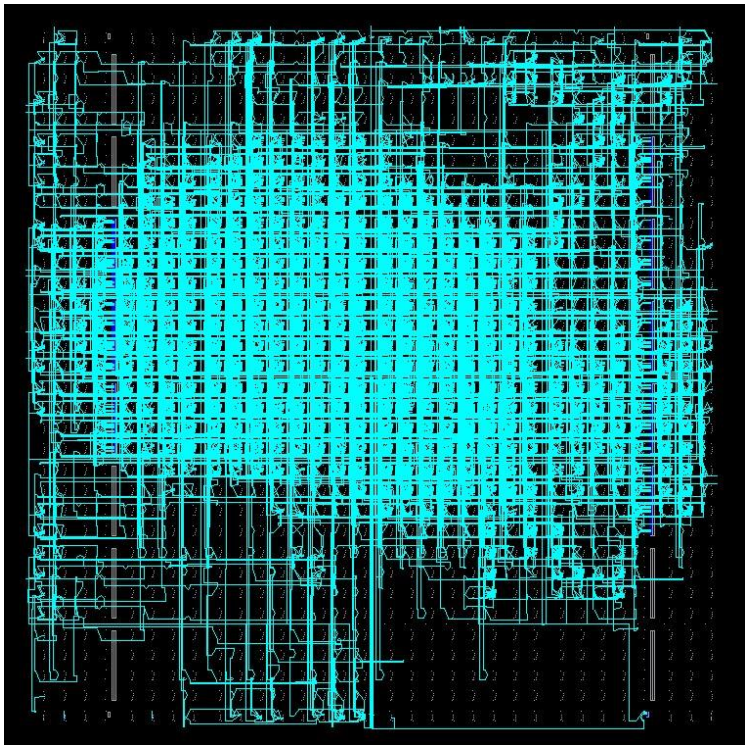
Network	Top-1 Error	Top-5 Error	Parameters	Compress Rate
LeNet-300-100 Ref	1.64%	-	1070 KB	
LeNet-300-100 Compressed	1.58%	-	**27 KB**	**40**×
LeNet-5 Ref	0.80%	-	1720 KB	
LeNet-5 Compressed	0.74%	-	**44 KB**	**39**×
AlexNet Ref	42.78%	19.73%	240 MB	
AlexNet Compressed	42.78%	19.70%	**6.9 MB**	**35**×
VGG-16 Ref	31.50%	11.32%	552 MB	
VGG-16 Compressed	31.17%	10.91%	**11.3 MB**	**49**×

S. Han et al.: "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding." (2015)

You don't need 32/64 bit Floating Point Numbers for Inference in Deep Neural Networks!

Quantization in Deep Learning

Particularly Interesting Cases of Quantization in DNNs are:



www-user.rhrk.uni-kl.de

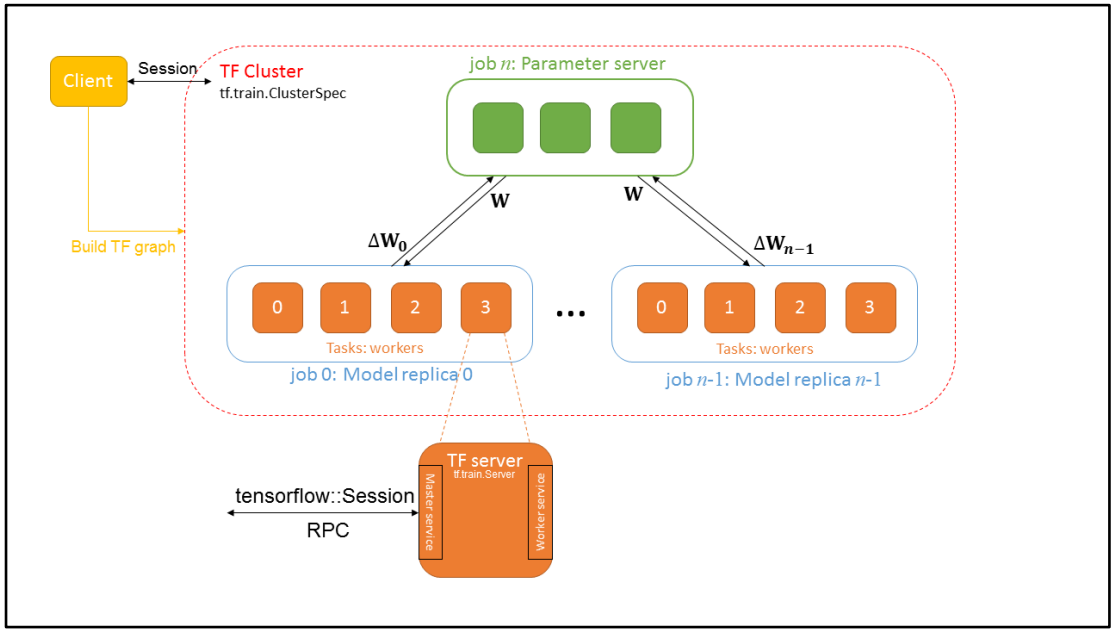
1) Custom Hardware in FPGAs and ASICs

ZDNet

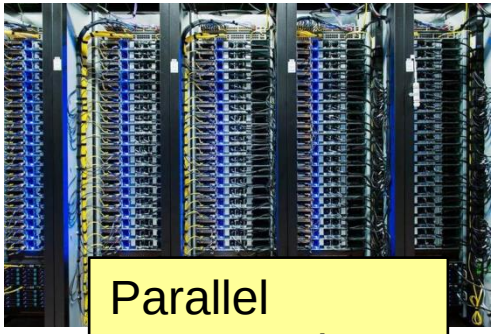
2) Communication in Distributed Systems

Fraunhofer ITWM

Quantization in Deep Learning



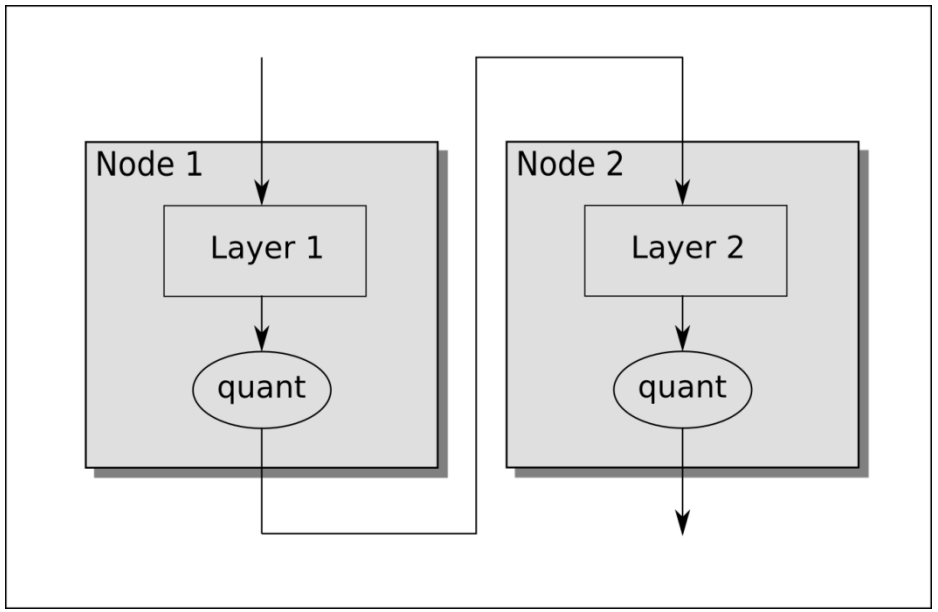
http://www.pittnuts.com/2016/08/glossary-in-distributed-tensorflow/

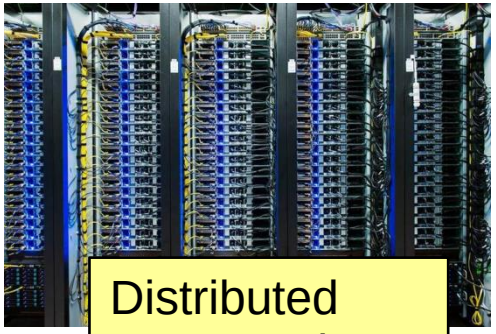


- There are several **copies of the same NN topology** on different nodes.
- All computations are carried out in **high precision** within the nodes.
- Updates for network parameters (e.g. gradients) are **quantized actively before they are sent** to another node.
- **Any quantization method** is applicable.

Fraunhofer ITWM

Quantization in Deep Learning

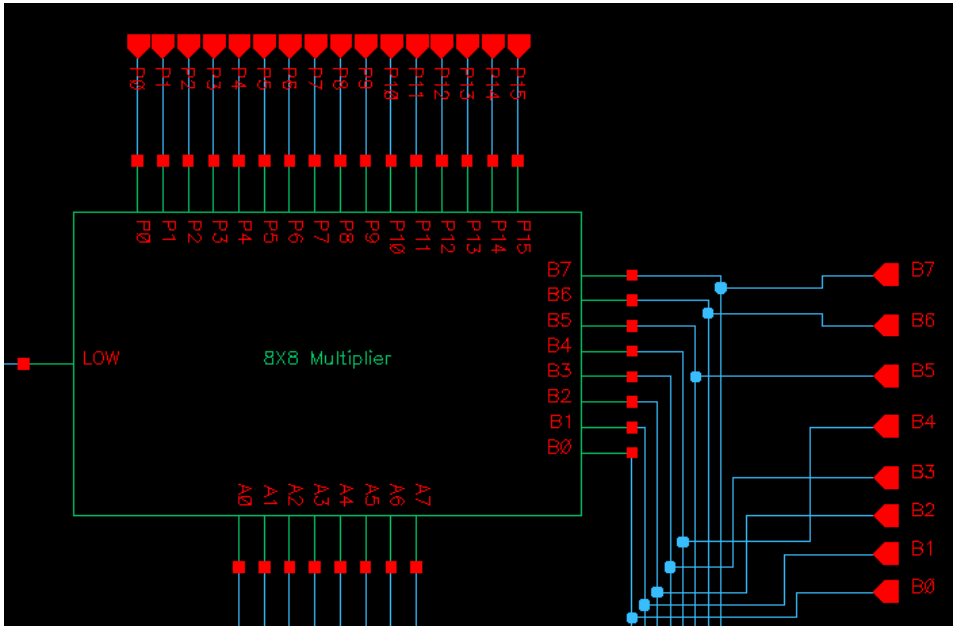




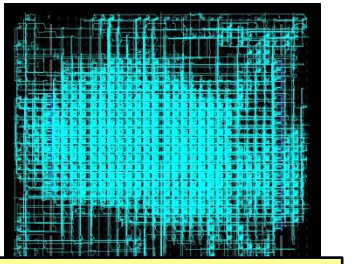
- The computation of a **single NN is distribured** on several nodes (e.g. each node computes a layer).
- All computations are carried out in **high precision** within the nodes.
- The results are **quantized actively before they are sent** to the next node.
- **Any quantization method** is applicable.

Fraunhofer ITWM

Quantization in Deep Learning



sites.tufts.edu

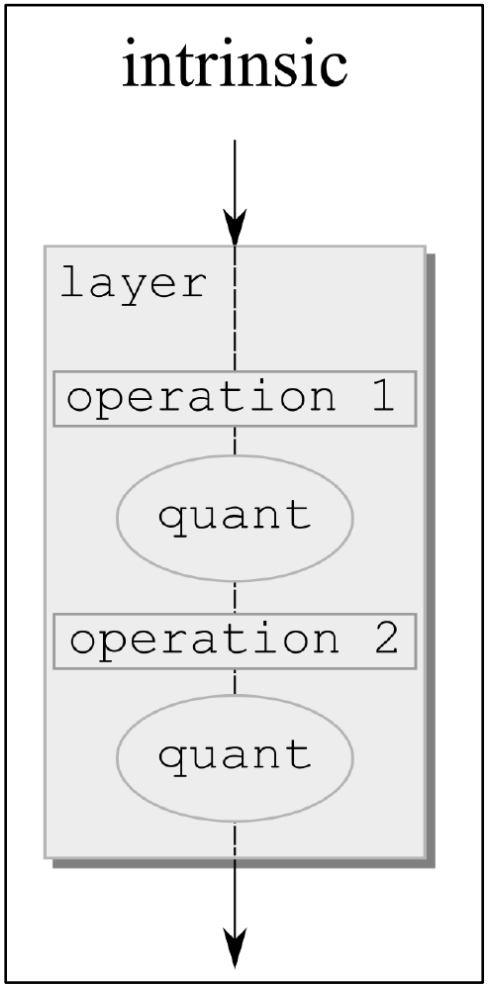


Hardware Accelerator

- Customizable hardware (e.g. FPGA or ASIC) **accelerates the computation** of a single layer type.
- **Fixed-point format** can be used to store and process data.
- All computations are carried out in **low precision** within the hardware.
- Quantization **happens passively after every single operation** (i.e. addition, multiplication, etc.).

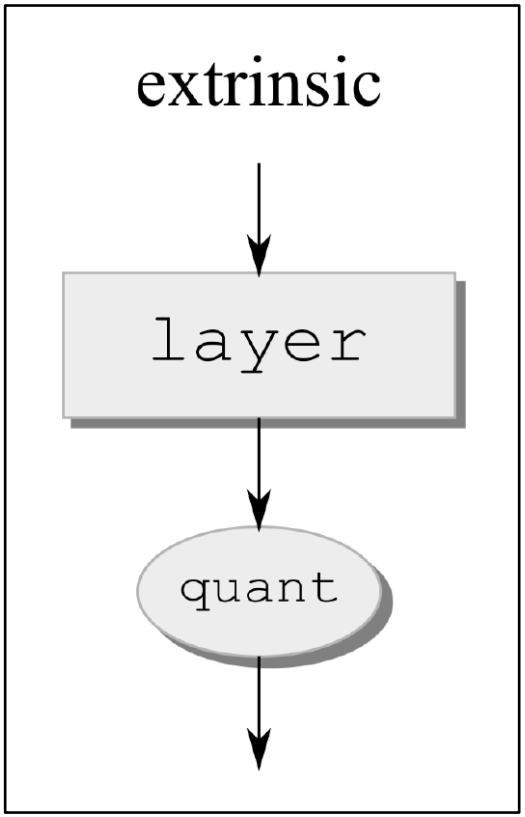
Fraunhofer ITWM

Quantization in Deep Learning



Intrinsic: **Emulate Custom Hardware**, where numbers are in fixed-point format. Quantize after every operation (i.e. addition, multiplication, etc.).

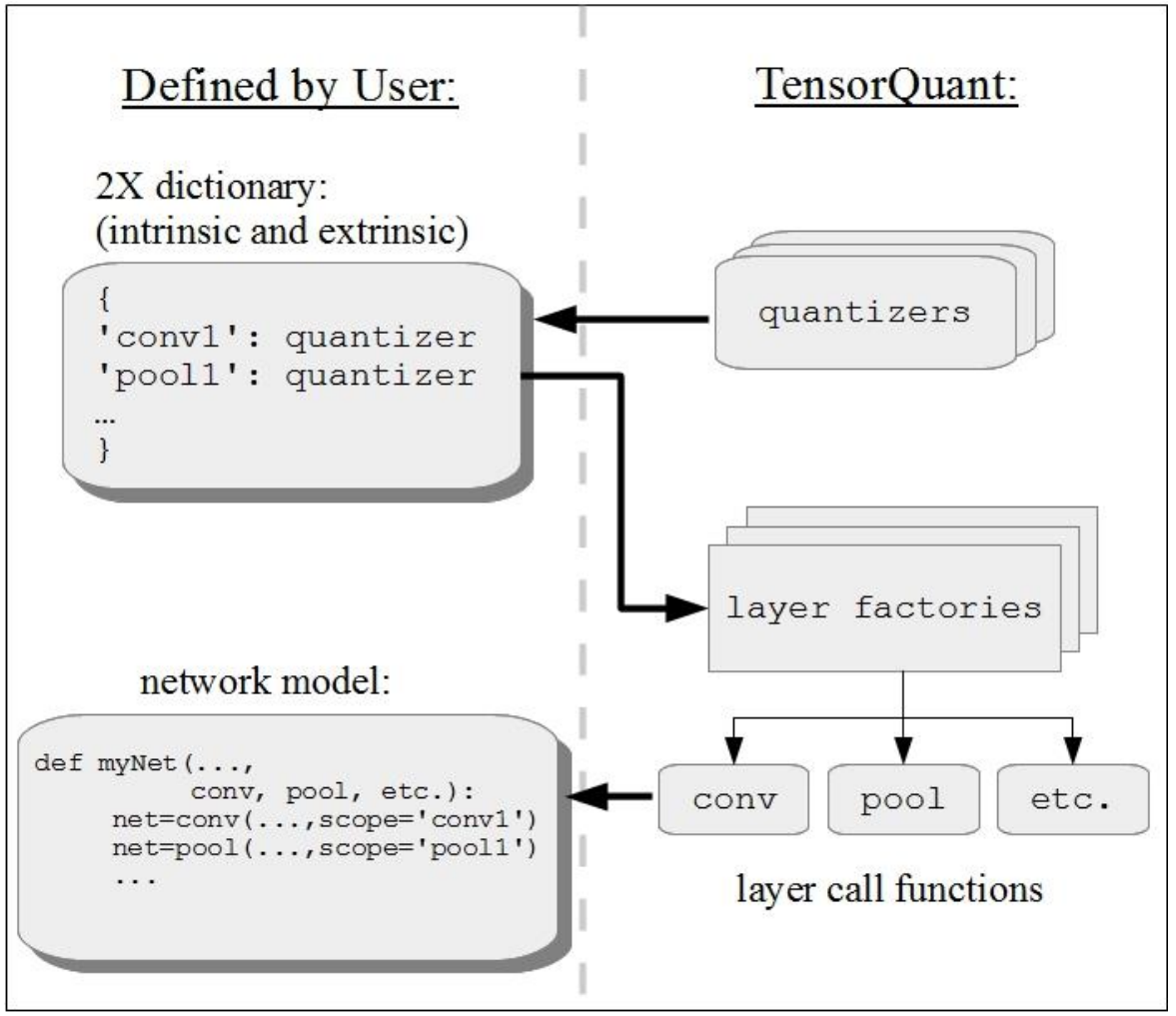
Extrinsic: **Simulates Distributed Systems,** where results are sent in quantized form. Quantize the results from the layer output.



Intrinsic and Extrinsic Quantization need to be distinguished!

Fraunhofer ITWM

The TensorQuant Toolbox



- Toolbox for TensorFlow
- Use any Quantization Method in Combination with your Topology
- Quantization during Training and Inference
- Quantization on various levels (intrinsic and extrinsic)

https://github.com/cc-hpc-itwm/TensorQuant

Fraunhofer ITWM

The TensorQuant Toolbox

Original Network Model:

```
def lenet(images, ...):
    ...
    with tf.variable_scope(scope, 'LeNet',
    [images, num_classes], ...):
    net = slim.conv2d(images, 32, [5, 5], scope='conv1')
    net = slim.max_pool2d(net, [2, 2], 2, scope='pool1')
    net = slim.conv2d(net, 64, [5, 5], scope='conv2')
    ...
    logits = slim.fully_connected(net, num_classes,
    activation_fn=None, scope='fc4')
    ...
    return logits, end_points
```

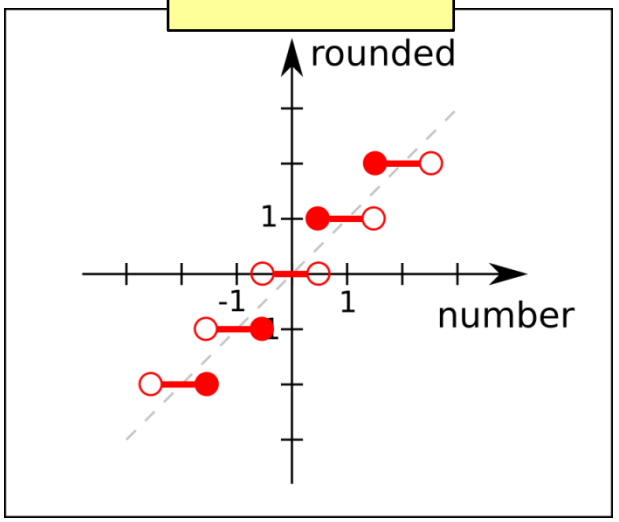
Prepared for TensorQuant:

```
def lenet(images, ...,
    conv2d=slim.conv2d,  # added layer types
    max_pool2d=slim.max_pool2d,
    fully_connected = slim.fully_connected):
    ...
    with tf.variable_scope(scope, 'LeNet',
    [images, num_classes],...):
    # removed slim. before layer calls
    net = conv2d(images, 32, [5, 5], scope='conv1')
    net = max_pool2d(net, [2, 2], 2, scope='pool1')
    net = conv2d(net, 64, [5, 5], scope='conv2')
    ...
    logits = fully_connected(net, num_classes,
    activation_fn=None, scope='fc4')
    ...
    return logits, end_points
```


Fraunhofer ITWM

Fixed-Point Experiments: Rounding

nearest



The common rounding

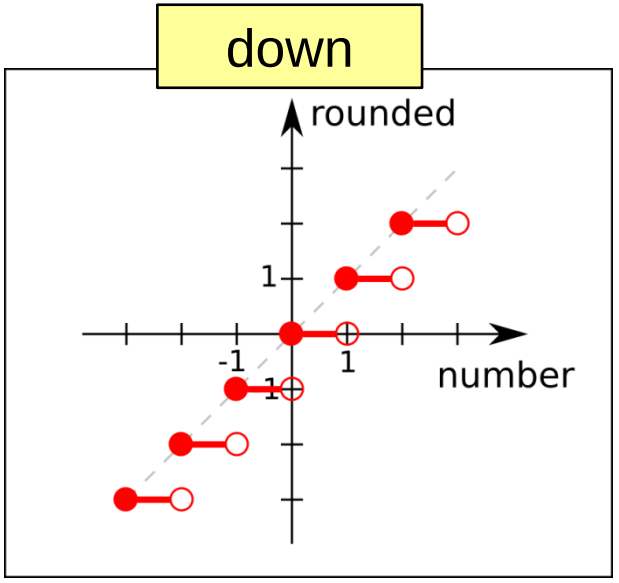
stochastic

$$\lceil \tfrac{x}{\Delta} \rceil \Delta \quad \text{if} \quad \tfrac{x}{\Delta} - \lfloor \tfrac{x}{\Delta} \rfloor \geqslant t_{\text{random}}$$

$$\lfloor \tfrac{x}{\Delta} \rfloor \Delta \quad \text{otherwise}$$

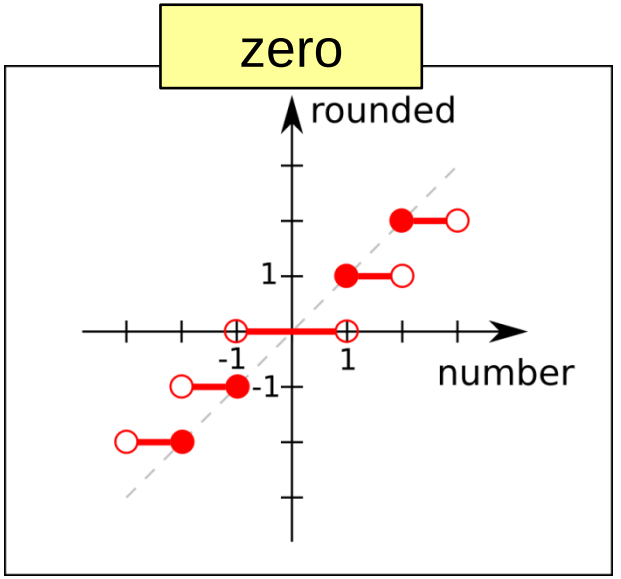
Like nearest, but sometimes rounded towards the opposite side

down



Cutting off the fractional part in two's complement

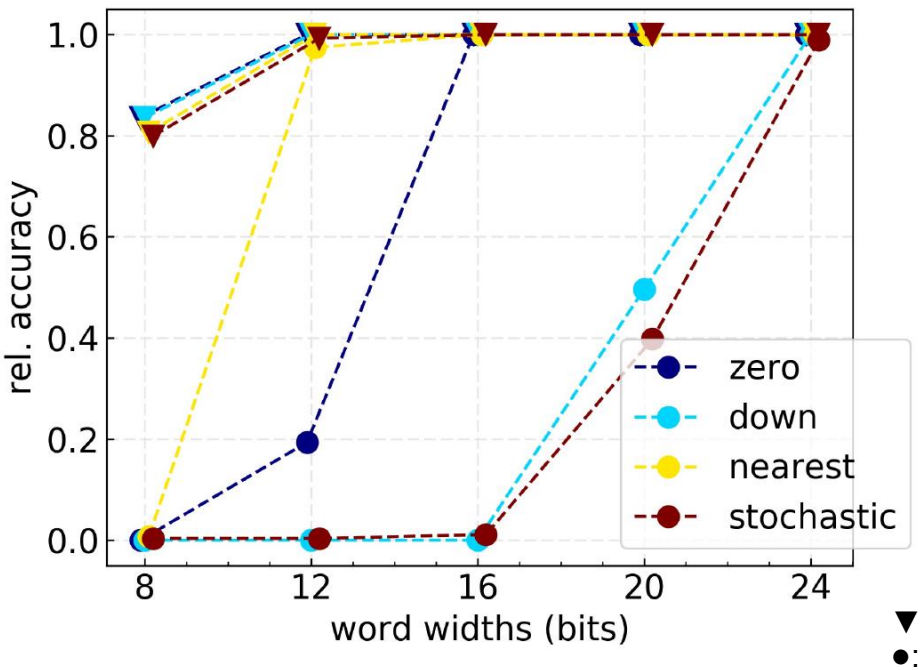
zero



Cutting off the fractional part in decimal representation

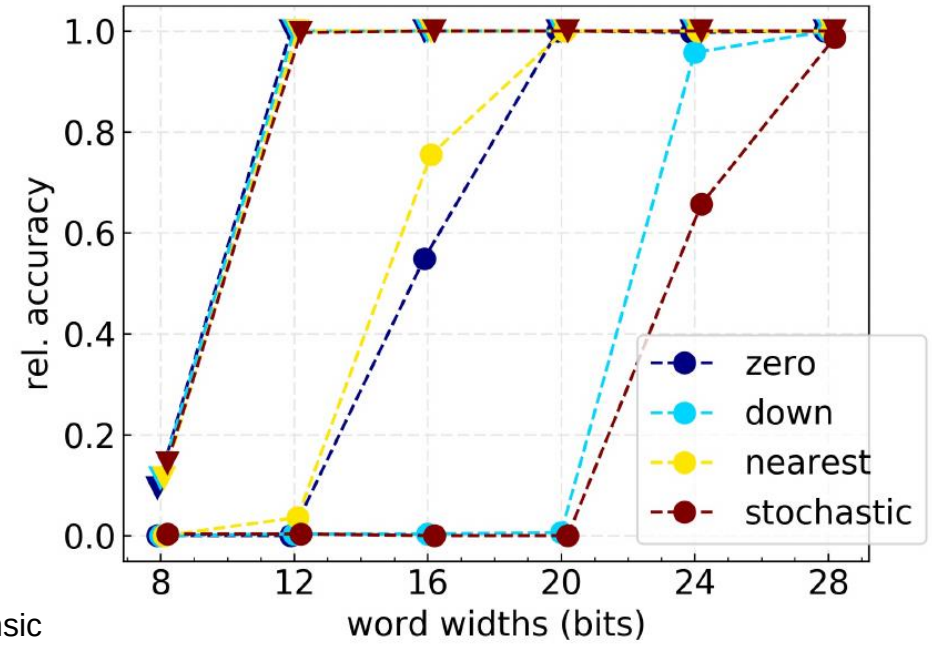
Fraunhofer ITWM

Fixed-Point Experiments: Rounding



GoogleNet

▼: extrinsic
●: intrinsic



ResNet50

The results are very different for intrinsic and extrinsic rounding!

rel. accuracy over fixed-point word width. $F = W/2$; rel. accuracy to pre-trained model without quantization

Fraunhofer ITWM

Fixed-Point Experiments: Extrinsic vs. Intrinsic

extrinsic:

Topology	4 bit	6 bit	8 bit	10 bit	12 bit
Inception V1	0.32, 0	0.76, 2	0.98, 2	1.00, 4	1.00, 4
Inception V3	0.08, 0	0.93, 2	0.97, 2	0.99, 4	1.00, 6
ResNet 50	0.06, 0	0.62, 0	0.99, 2	1.00, 4	1.00, 4
ResNet 152	0.07, 0	0.55, 0	1.00, 2	1.00, 2	1.00, 2

intrinsic:

Topology	8 bit	10 bit	12 bit	14 bit	16 bit
Inception V1	0.46, 2	0.98, 5	1.00, 8	1.00, 9	1.00, 8
Inception V3	0.08, 2	0.92, 7	0.97, 6	0.99, 7	0.98, 6
ResNet 50	0.02, 2	0.14, 3	0.54, 4	0.93, 5	0.99, 6
ResNet 152	0.03, 2	0.16, 3	0.55, 4	0.95, 5	1.00, 6

Fixed-point word widths applied to various topologies. Entries are (rel. accuracy, fractional bits *F*)

Fixed-Point Experiments: LeNet

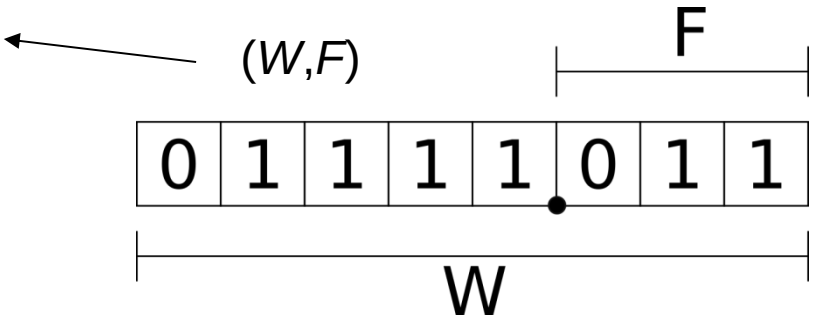
Topology	4 bit	8 bit	12 bit	16 bit	20 bit
LeNet	0.40, 3	0.90, 4	1.00, 6	1.00, 8	1.00, 8
Inception V1	-	0.01, 4	0.99, 6	1.00, 8	1.00, 8
ResNet 50	-	0.00, 0	0.04, 6	0.91, 6	1.00, 10

intrinsic, nearest rounding; entries are (rel. accuracy,*F*); best *F* for given *W*.

Fixed-Point Experiments: Layers

Inception V3

Subunit	100%	99%	90%
Conv2d_1a_3x3	24,6	16,12	12,6
Conv2d_2a_3x3	16,6	12,6	8,2
Conv2d_2b_3x3	12,6	12,6	8,4
MaxPool_3a_3x3	8,4	8,4	8,0
Conv2d_3b_1x1	12,6	12,6	8,4
Conv2d_4a_3x3	8,4	8,4	8,4
MaxPool_5a_3x3	8,0	8,0	8,0
Mixed_5b	8,4	8,4	8,4
Mixed_5c	12,4	12,4	8,4
Mixed_5d	16,10	16,10	8,4
Mixed_6a	8,4	8,4	8,4
Mixed_6b	12,6	12,6	8,4
Mixed_6c	12,6	12,6	8,6
Mixed_6d	12,8	12,8	12,6
Mixed_6e	12,8	12,8	12,6
Mixed_7a	24,20	16,8	8,4
Mixed_7b	8,6	8,6	8,6
Mixed_7c	8,6	8,6	8,6
AuxLogits	8,0	8,0	8,0
PreLogits	8,0	8,0	8,0
Logits	12,4	12,4	12,4
compression:	x3.1	x3.2	x3.6



intrinsic, nearest rounding; entries are the smallest combination (*W*,*F*) to achieve rel. accuracy stated in column header

Fraunhofer ITWM

Plans for TensorQuant

- Quantized Training
  - Quantize the Gradient
  - Quantize the Application of the Gradient to the Variables
- Other Quantization Methods
  - Logarithmic
  - Clustering
- Quantized RNNs
- Binary / Ternary DNNs

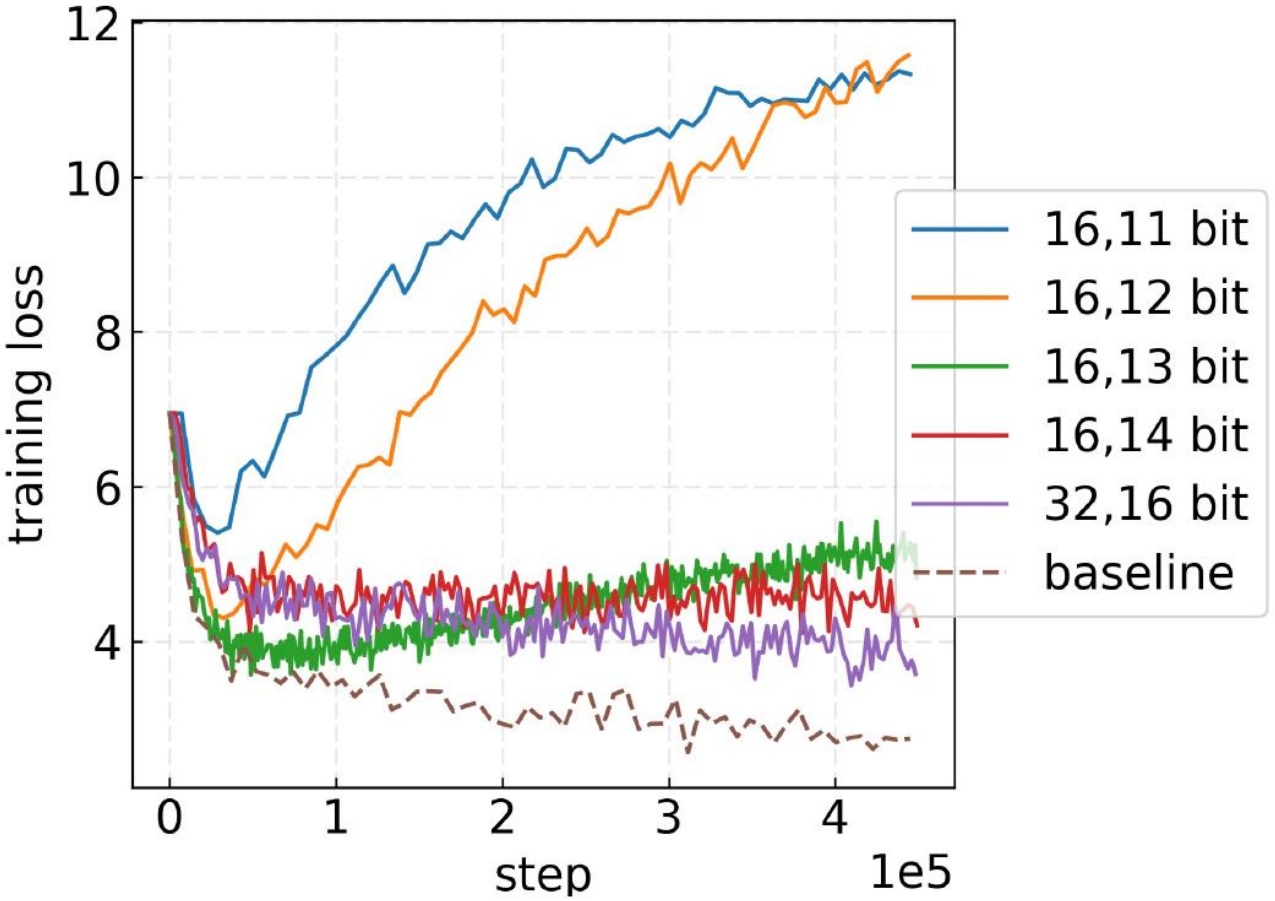
Take-Home Messages

- It matters not only HOW, but also WHERE the network is quantized.
- Do not transfer conclusions from one topology to another.
- TensorQuant is a Toolbox for TensorFlow, which helps to investigate quantization on any level.

dominik.loroch@itwm.fhg.de
https://github.com/cc-hpc-itwm/TensorQuant
booth 2143

Thank You!

Appendix: Quantized Gradient



AlexNet (quantized gradient)

(W,F)	16,8	16,11	16,12	16,13	16,14	32,16
rel. accuracy	0.004	0.847	0.926	0.997	0.745	1.0

Fraunhofer ITWM