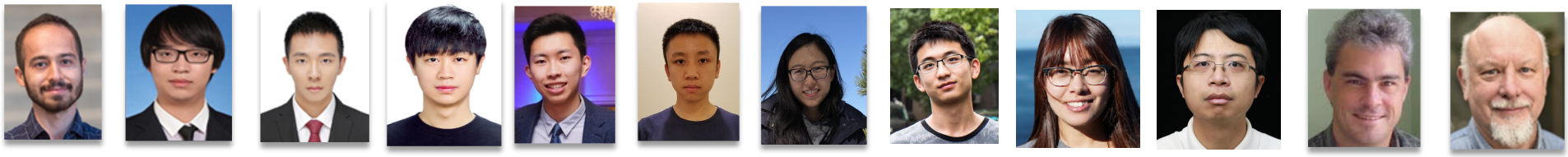
Systematic Neural Network Quantization

Amir Gholami, Z. Yao, Z. Dong, S. Kim, Z. Zheng, E. Tan, Q. Huang, S. Yu, L. Wang, Y. Wang, Michael Mahoney, Kurt Keutzer

University of California Berkeley, in collaboration with AWS

GTC Conference, April, 2021



Quantization: Workhorse for Efficient Inference

- Uniform quantization is a mapping from floating point values to quantized integer values

0.34	3.75	5.64
1.12	2.7	-0.9
-4.7	0.68	1.43

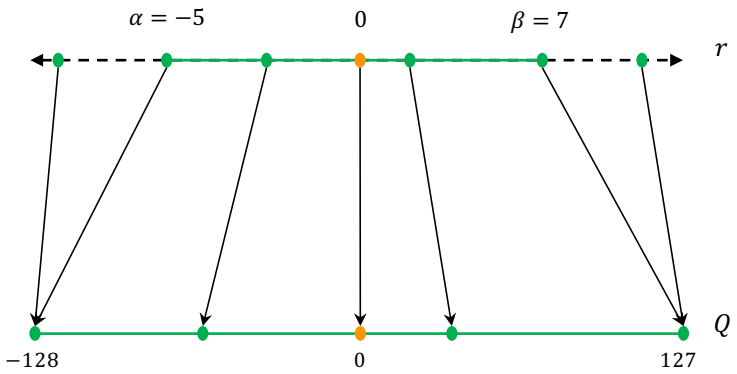
FP32
(pre-quantized)

64	134	217
76	119	21
3	81	99

INT8
(quantized)

Illustration from Sahni Manas

Floating Point Values



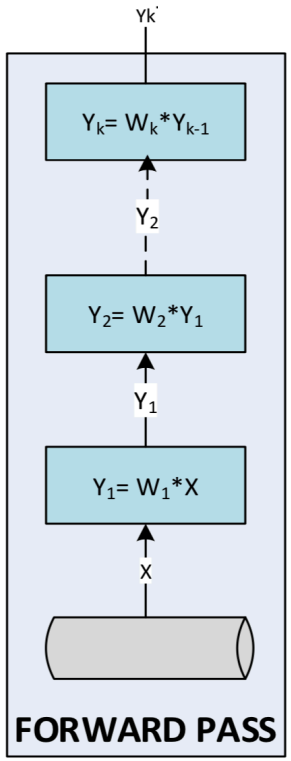
8-bit Quantized Values

$$Q(r) = \text{Int}\left(r/S\right) - Z,$$

Gholami A, Kim S, Dong Z, Yao Z, Mahoney MW, Keutzer K. A Survey of Quantization Methods for Efficient Neural Network Inference. arXiv preprint arXiv:2103.13630. 2021.

Quantization

Let's take a closer look at a layer



$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \\ w_{20} & w_{21} & w_{22} \\ w_{30} & w_{31} & w_{32} \\ w_{40} & w_{41} & w_{42} \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}$$

$$\boldsymbol{y} = W \cdot \boldsymbol{x}$$

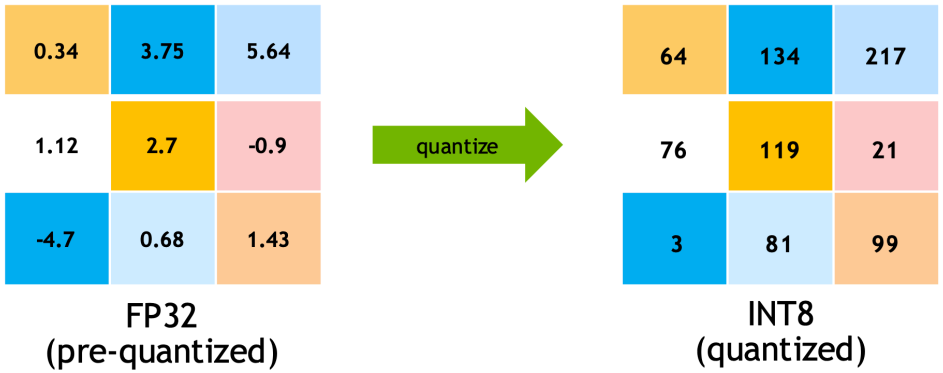


Illustration from Sahni Manas

Closer Look at One Layer

For simplicity, let's consider symmetric quantization => Z=0

We first need to accumulate results in INT32 and then rescale back to INT4

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \\ w_{20} & w_{21} & w_{22} \\ w_{30} & w_{31} & w_{32} \\ w_{40} & w_{41} & w_{42} \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}$$

$$\boldsymbol{y} = W \cdot \boldsymbol{x}$$

$$(S_y, y^{i4}) = (S_w, W^{i4}) \cdot (S_x, x^{i4})$$

$$y^{i32} = W^{i4} x^{i4}$$

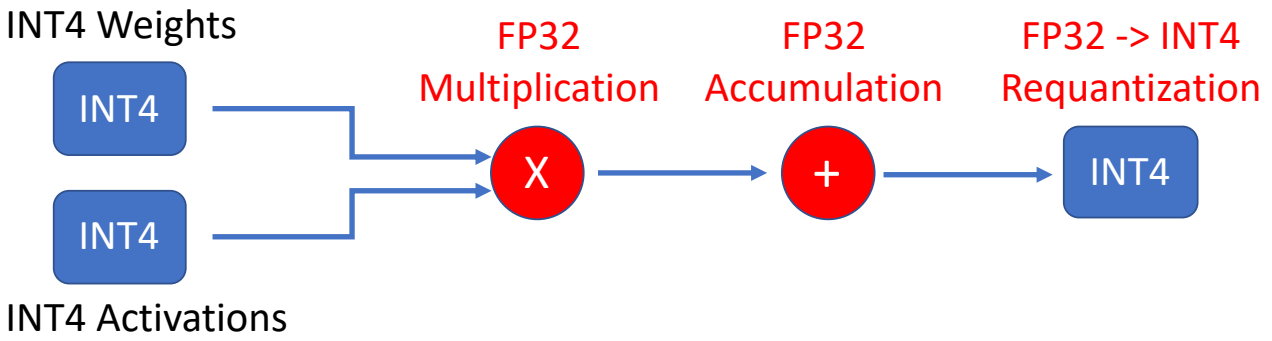
$$y^{i4} = Round(\frac{S_x S_w}{S_y} y^{i32})$$

This is Integer-only (aka fixed-point/Dyadic quantization)

But not all quantization algorithms use this method

Simulated Quantization!

Simulated Quantization: (fake quantization)



- Simulated quantization performs arithmetic in software using **FP32** arithmetic, but this can lead to discrepancy when deployed on integer-only hardware
- Also many components such as **BN** or **residual connection** are computed using **FP32** which is not executable on integer-only hardware

Z. Yao*, Z. Dong*, Z. Zheng*, A. Gholami*, E. Tan, J. Li, L. Yuan, Q. Huang, Y. Wang, M. W. Mahoney, K. Keutzer, HAWQ-V3: Dyadic Neural Network Quantization in Mixed Precision, arxiv:2011.10680, 2020.

Simulated/Fake Quantization Error

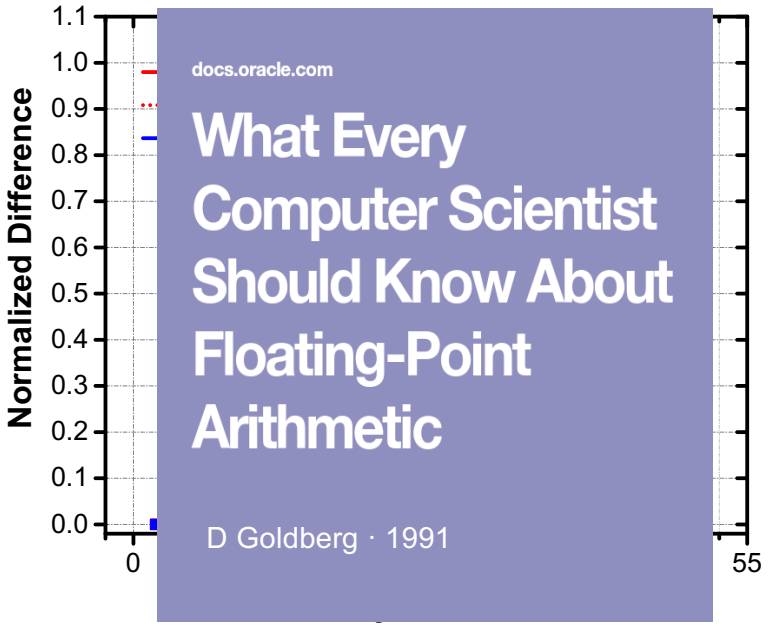
- Simulated/Fake quantization can create O(1) error for simple operations such as residual connection. This is because rounding operation is not linear:

$$Int(a+b) \neq Int(a) + Int(b)$$

Example:

$$Int(2.4+1.3) = 4 \neq Int(2.4) + Int(1.3) = 3$$

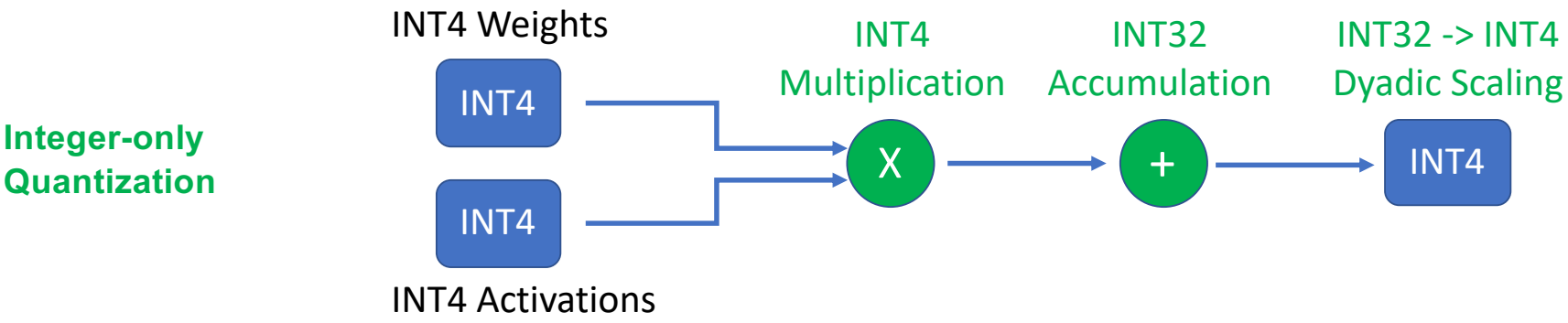
- This difference is O(1) and will propagate throughout the network, especially for low precision quantization as shown in the figure



The normalized difference between fake quantization in Pytorch and corresponding values when deployed in hardware for ResNet50 on ImageNet. As one can see the error becomes quite significant, especially for low precision quantization

Z. Yao*, Z. Dong*, Z. Zheng*, A. Gholami*, E. Tan, J. Li, L. Yuan, Q. Huang, Y. Wang, M. W. Mahoney, K. Keutzer, HAWQ-V3: Dyadic Neural Network Quantization in Mixed Precision, arxiv:2011.10680, 2020.

Integer-only Quantization!



- With integer only quantization we perform all the arithmetic using INT multiplication and addition, without any FP calculation for the entire inference. This includes:
  - CONV, **BN**, Residual Connection, Pooling, and even non-linear activation
  - ➢ **BN:** Need to first fine-tune BN statistics using high precision and after correctly computing the statistics it can be quantized and fused with conv layer

Z. Yao*, Z. Dong*, Z. Zheng*, A. Gholami*, E. Tan, J. Li, L. Yuan, Q. Huang, Y. Wang, M. W. Mahoney, K. Keutzer, HAWQ-V3: Dyadic Neural Network Quantization in Mixed Precision, arxiv:2011.10680, 2020.

Integer-only Quantization Works: CV

(a) *MobileNetV2*

Method	Int	Uni	BL	Precision	Size	BOPS	Top-1
Baseline	✗	✓	73.03	W32A32	13.2	322	73.03
RVQuant (Park et al., 2018)	✗	✗	70.10	W8A8	3.3	20	70.29
CalibTIB(Hubara et al., 2020)	✗	✓	71.90	W8A8	3.3	20	71.60
HAWQV3	✓	✓	73.03	W8A8	3.3	20	**73.01**

(b) *ResNet50*

Method	Int	Uni	BL	Precision	Size	BOPS	Top-1
Baseline	✓	✓	77.72	W32A32	97.8	3951	77.72
Integer Only (Jacob et al., 2018)	✓	✓	76.40	W8A8	24.5	247	74.90
RVQuant (Park et al., 2018)	✗	✗	75.92	W8A8	24.5	247	75.67
HAWQV3	✓	✓	77.72	W8A8	24.5	247	**77.58**

(c) *InceptionV3*

Method	Int	Uni	BL	Precision	Size	BOPS	Top-1
Baseline	✗	✓	78.88	W32A32	90.9	5850	78.88
Integer Only (Jacob et al., 2018)	✓	✓	78.30	W8A8	22.7	366	74.20
RVQuant (Park et al., 2018)	✗	✗	74.19	W8A8	22.7	366	74.22
HAWQV3	✓	✓	78.88	W8A8	22.7	366	**78.76**

Z. Yao*, Z. Dong*, Z. Zheng*, A. Gholami*, E. Tan, J. Li, L. Yuan, Q. Huang, Y. Wang, M. W. Mahoney, K. Keutzer, HAWQ-V3: Dyadic Neural Network Quantization in Mixed Precision, arxiv:2011.10680, 2020.

Integer-only Quantization Works: Tranformers

(a) RoBERTa-Base

	Int-only	MNLI-m	MNLI-mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
Baseline	✗	**87.8**	**87.4**	**90.4**	**92.8**	94.6	61.2	**91.1**	90.9	78.0	86.0
I-BERT	✓	87.5	**87.4**	90.2	**92.8**	**95.2**	**62.5**	90.8	**91.1**	**79.4**	**86.3**
Diff		-0.3	0.0	-0.2	0.0	+0.6	+1.3	-0.3	+0.2	+1.4	+0.3

(b) RoBERTa-Large

	Int-only	MNLI-m	MNLI-mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
Baseline	✗	90.0	89.9	92.8	94.1	96.3	68.0	**92.2**	91.8	86.3	89.0
I-BERT	✓	**90.4**	**90.3**	**93.0**	**94.5**	**96.4**	**69.0**	**92.2**	**93.0**	**87.0**	**89.5**
Diff		+0.4	+0.4	+0.2	+0.4	+0.1	+1.0	0.0	+1.2	+0.7	+0.5

S. Kim*, A. Gholami*, Z. Yao*, M. Mahoney, K. Keutzer, **I-BERT: Integer-only BERT Quantization**, arxiv:2101.01321, 2021. [Online Code]

Integer-only Quantization Works: ASR

(a) *QuartzNet-15x5*

Method	W/A	dev		test		Size (MB)	BOPs (G)
		clean	other	clean	other		
Baseline	32/32	3.80	10.05	3.82	10.08	73.81	1890
Q-ASR	8/8	3.92	10.28	4.04	10.37	18.45	118

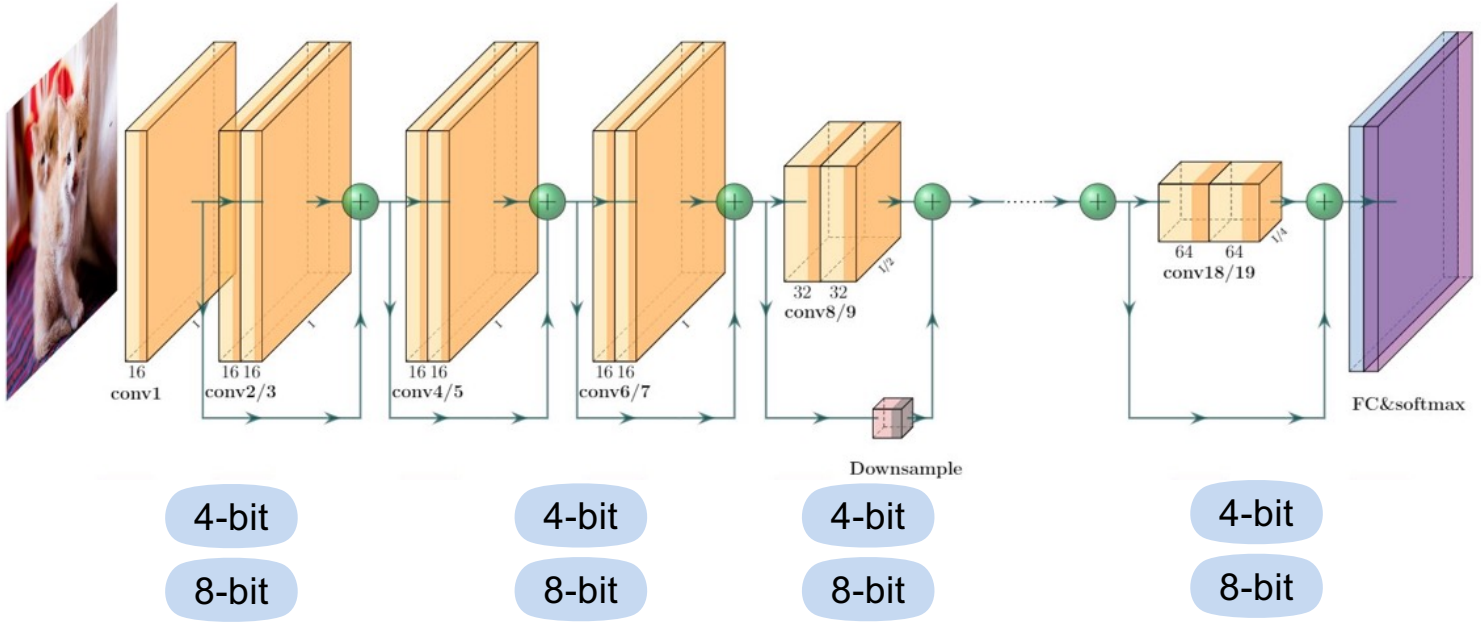
(b) *JasperDr-10x5*

Method	W/A	dev		test		Size (MB)	BOPs (T)
		clean	other	clean	other		
Baseline	32/32	3.47	10.40	3.68	10.49	1208	33.3
Q-ASR	8/8	3.47	10.49	3.68	10.57	302.0	2.08

S. Kim, A. Gholami, Z. Yao, A. Nrusimha, B. Zhai, T. Gao, M. Mahoney, K. Keutzer, **Q-ASR: Integer-only Zero-shot Quantization for Efficient Speech Recognition** arxiv: 2103.16827, 2021. [Online Code]

What about Sub-INT8 Quantization?

Can we go even further and perform lower precision quantization?



Uniform low precision does not work as it can significantly degrade accuracy => **Use mixed-precision**

- But how should we set the precision for each kernel?

Hessian Aware Quantization

This is somewhat similar to the **Jenga** game. We only remove blocks that are not sensitive.

- Only use low precision quantization for insensitive parameters (flat loss landscape)
- Use high precision quantization for sensitive parameters (sharp loss landscape)

This sensitivity can be calculated through Hessian which quantifies the relative sharpness/flatness of the loss landscape.

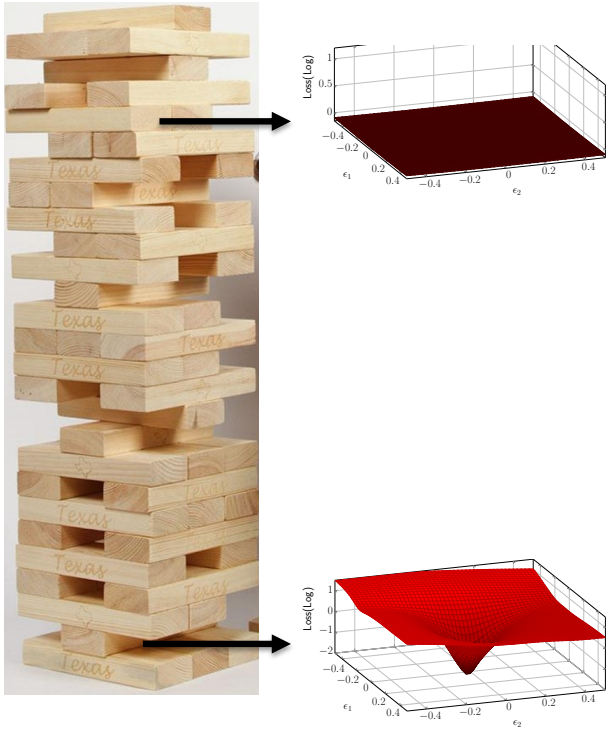
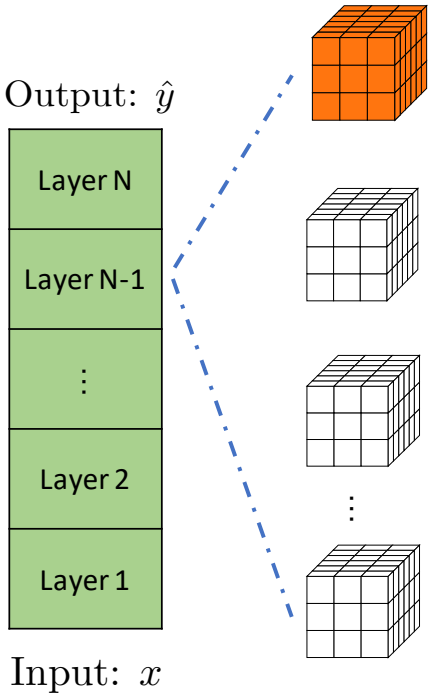


Image from UniversityCoop

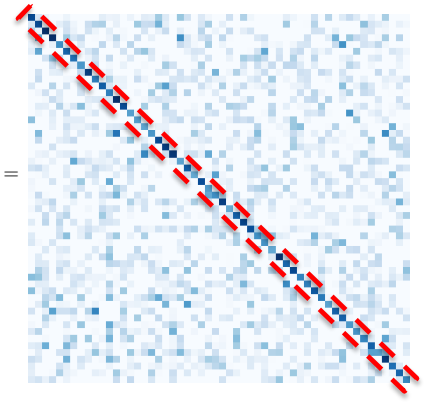


Yu S, Yao Z, Gholami A, Dong Z, Mahoney MW, Keutzer K. Hessian-Aware Pruning and Optimal Neural Implant. arXiv preprint arXiv:2101.08940.
Dong Z, Yao Z, Cai Y, Arfeen D, Gholami A, Mahoney MW, Keutzer K. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. NeurIPS, 2020.

Hessian Trace can Quantify Sharpness/Flatness

Theorem 1 *After quantizing the model to same precision, fine tuning layers that have smaller average trace of Hessian can achieve a smaller loss, compared to layers with larger* ***average trace of Hessian****.*

$$\text{average } tr(H) = \frac{1}{n}\sum_i H_{ii} = \frac{1}{n}\sum_i \lambda_i(H)$$

$Diag(H) =$ 

- Relatively large Hessian trace => sharp loss landscape => Use high precision
- Relatively **small Hessian trace** => flat loss landscape => Use **low precision**

Dong Z, Yao Z, Cai Y, Arfeen D, Gholami A, Mahoney MW, Keutzer K. HAWQ-V2: Hessian aware trace-weighted quantization of neural networks. NeurIPS, 2020.

Results: ResNet50

(b) *ResNet50*

Method	Int	Uni	BL	Precision	Size	BOPS	Top-1
Baseline	✓	✓	77.72	W32A32	97.8	3951	77.72
Integer Only (Jacob et al., 2018)	✓	✓	76.40	W8A8	24.5	247	74.90
RVQuant (Park et al., 2018)	✗	✗	75.92	W8A8	24.5	247	75.67
HAWQV3	✓	✓	77.72	W8A8	24.5	247	**77.58**
PACT (Choi et al., 2018)	✗	✓	76.90	W5A5	16.0	101	76.70
LQ-Nets (Zhang et al., 2018)	✗	✗	76.50	W4A32	13.1	486	76.40
RVQuant (Park et al., 2018)	✗	✗	75.92	W5A5	16.0	101	75.60
HAQ (Wang et al., 2019)	✗	✗	76.15	WMPA32	9.62	520	75.48
OneBitwidth (Chin et al., 2020)	✗	✓	76.70	W1*A8	12.3	494	76.70
HAWQV3	✓	✓	77.72	W4/8A4/8	18.7	154	75.39
HAWQV3+DIST	✓	✓	77.72	W4/8A4/8	18.7	154	**76.73**

Z. Yao*, Z. Dong*, Z. Zheng*, A. Gholami*, E. Tan, J. Li, L. Yuan, Q. Huang, Y. Wang, M. W. Mahoney, K. Keutzer, HAWQ-V3: Dyadic Neural Network Quantization in Mixed Precision, arxiv:2011.10680, 2020.

https://github.com/zhen-dong/hawq

HAWQ Overhead?

Hessian AWare Quantization Timing on a 4-Titan RTX system:

- All the calculations, including Hessian, for bit selection takes **less than 1-hour**
- Up to **120x faster than AutoML** based methods, despite using Hessian, with **higher accuracy**

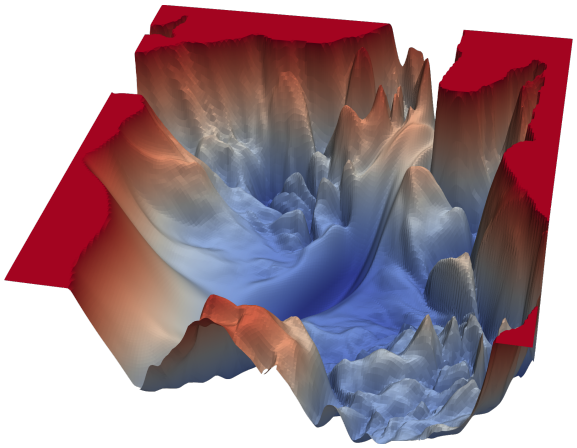
Comparison between HAWQ-V2 and HAQ on ResNet50, InceptionV3 and SqueezeNext models.

Model	Method	Top-1 Accuracy	W-Comp	Size(MB)	Search Time(hours)	Speed Up
ResNet50	Baseline Model	**77.39**	1.00×	97.8		
ResNet50	HAQ	**75.30**	10.57×	9.22	**10**	
ResNet50	HAWQ-V2	**75.56**	**12.24**×	7.99	**0.5**	**> 20×**
InceptionV3	Baseline Model	**77.45**	1.00×	91.2		
InceptionV3	HAQ	**71.60**	10.00×	9.12	**50**	
InceptionV3	HAWQ-V2	**75.68**	**12.04**×	7.57	**0.4**	**> 125×**
SqueezeNext	Baseline Model	**69.38**	1.00×	10.1		
SqueezeNext	HAQ	**65.87**	10.00×	1.01	**50**	
SqueezeNext	HAWQ-V2	**68.38**	9.40×	1.07	**0.9**	**> 55×**

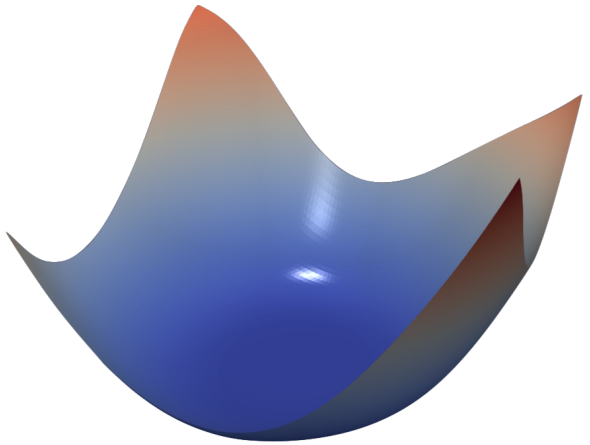
Dong Z, Yao Z, Cai Y, Arfeen D, Gholami A, Mahoney MW, Keutzer K. HAWQ-v2: Hessian aware trace-weighted quantization of neural networks. NeurIPS, 2020.

Summary:
One Size Does Not Fit All

- Quantization (and pruning) of a model depends on the loss landscape which is a function of:
  - Model Architecture
  - Dataset
- We can automatically quantify the sensitivity to quantization and pruning through Hessian



Less Robust to Quantization



More Robust to Quantization

Yu S, Yao Z, Gholami A, Dong Z, Mahoney MW, Keutzer K. Hessian-Aware Pruning and Optimal Neural Implant. arXiv preprint arXiv:2101.08940.
Yao Z, Gholami A, Shen S, Keutzer K, Mahoney MW. ADAHESSIAN: An adaptive second order optimizer for machine learning. AAAI, 2020.
Yao Z, Gholami A, Lei Q, Keutzer K, Mahoney MW. Hessian-based analysis of large batch training and robustness to adversaries. NeruIPS, 2018.
Yao Z, Gholami A, Xu P, Keutzer K, Mahoney MW. Trust region based adversarial attack on neural networks. CVPR, 2018.
Images from Li H et al.

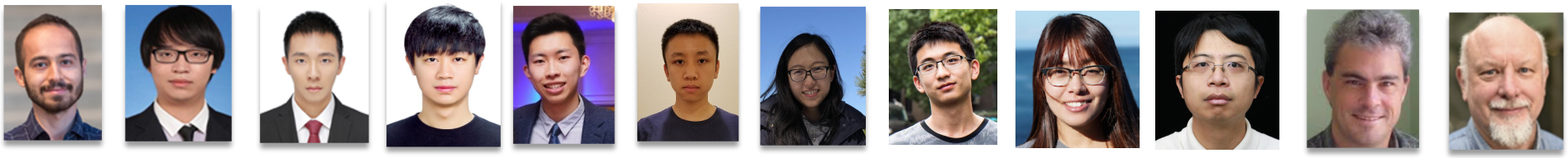
Summary

- **Integer-only quantization is possible without any accuracy degradation for INT8:**
  - Up to **4.5% higher accuracy compared to previous SOTA** from Google and close accuracy degradation gap with integer-only quantization
  - Works on compact models such as MobileNet (even with their strong baseline)
  - Works across different tasks: CV, ASR, and even NLP with I-BERT
- **Sub-INT8 quantization can be performed even without AutoML through HAWQv3,** a fully automatic framework without any manual bit selection
  - Key idea is to only perform sub-INT8 quantization for **insensitive layers**
    - Sensitivity can be efficiently calculated through Hessian
    - Up to **1.5x speed up with INT4/INT8 compared to INT8 quantization**
- **All results verified with direct, end-to-end hardware implementation, and open-sourced**

Thanks

Thank You for Listening!

amirgh@berkeley.edu



Berkeley
UNIVERSITY OF CALIFORNIA