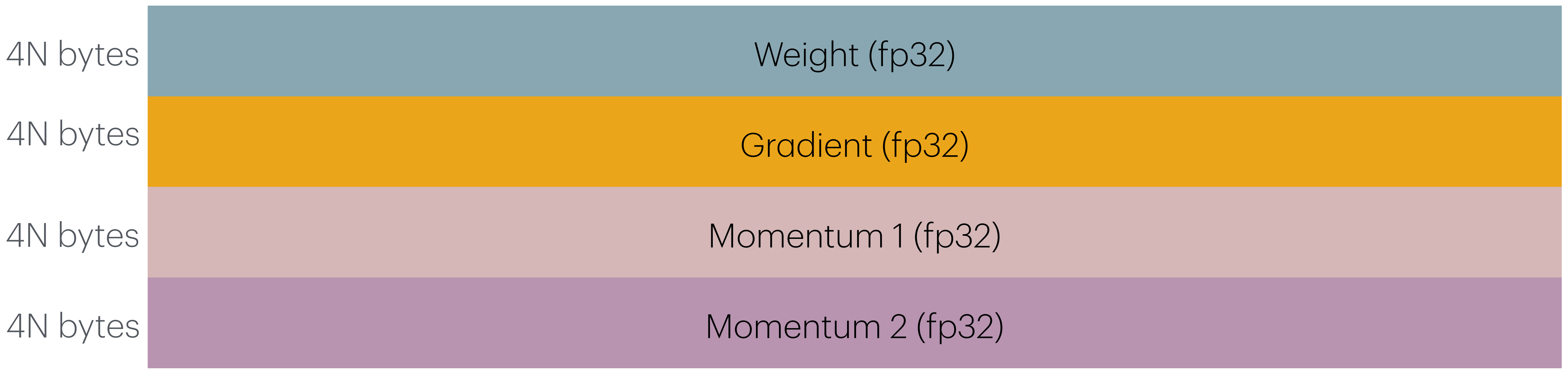
Quantization

Philipp Krähenbühl, UT Austin

Training large models

Memory requirements

- Without optimization
  - Model parameters: N
  - Weights: N floats
  - Gradients: N floats
  - Momentum: N floats
  - 2nd momentum (ADAM): N floats
- 16N bytes without counting activations



Inference in large models

4N bytes | Weight (fp32)

Memory requirements

- Without optimization
  - Model parameters: N
  - Weights: N floats
- 4N bytes without counting activations
- 8B parameter model (i.e. llama 3.1)
  - 32GB of memory for just weights

Model	Micro-architecture	Launch	Core	Memory				
				Bus type	Bus width (bit)	Size (GB)	Clock (MT/s)	Bandwidth (GB/s)
A100 GPU accelerator (PCIe card)[44][45]	Ampere	May 14, 2020[46]	1× GA100-883AA-A1	HBM2	5,120	40 or 80	1,215	1,555
A40 GPU accelerator (PCIe card)[43]	Ampere	October 5, 2020	1× GA102	GDDR6	384	48	7,248	695.8
L40 GPU accelerator[50]	Ada Lovelace	October 13, 2022	1× AD102[51]	GDDR6	384	48	2,250	864
H100 GPU accelerator (PCIe card)[47]	Hopper	March 22, 2022[48]	1× GH100[49]	HBM2E	5120	80	1,000	2,039
H100 GPU accelerator (SXM card)	Hopper	March 22, 2022[48]	1× GH100[49]	HBM3	5,120	80	1,500	3,352

[1] https://en.wikipedia.org/wiki/List_of_Nvidia_graphics_processing_units#Data_Center_GPUs

Inference in large models

2N bytes | Weight (bf16)

Memory requirements

- Without optimization
  - Model parameters: N
  - Weights: N bfloat16
- 4N bytes without counting activations
- 8B parameter model (i.e. llama 3.1)
  - 16GB of memory for just weights

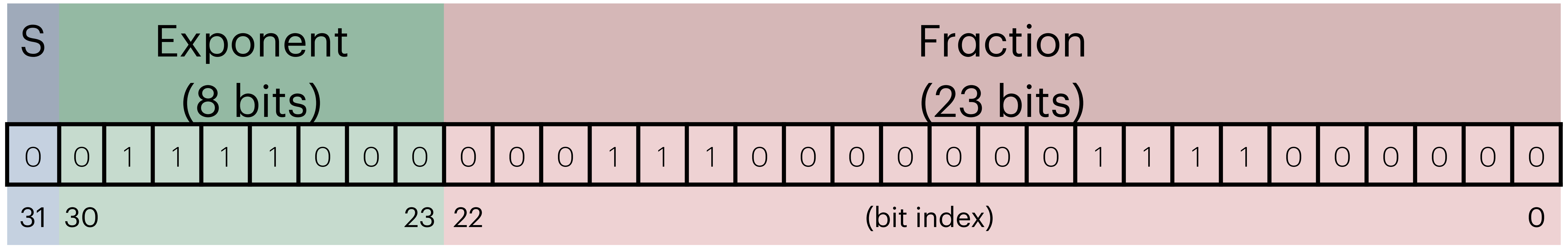
Image credit: Wikipedia

float32

float32	
Sign	1 bit
Exponent	8 bit
Mantissa	23 bit
Precision (relative)	1E-07
Max value	3E+38
Min value (normal)	1.7E-38

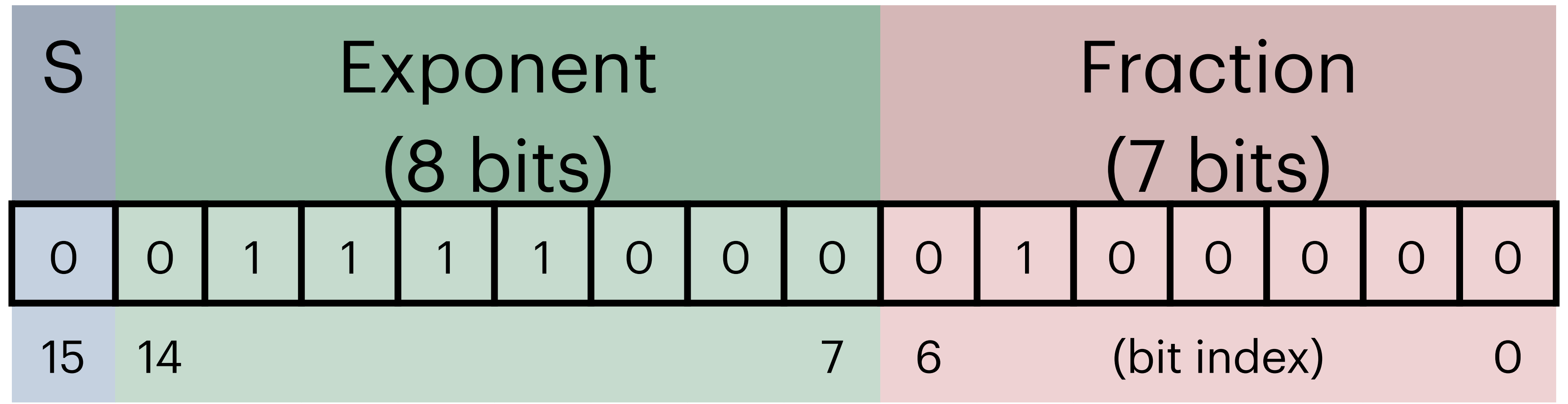
Precision (relative) = $\dfrac{x_2 - x_1}{x_1}$

where x_2 is smallest value $x_2 > x_1$



Bfloat16

Bfloat16	
Sign	1 bit
Exponent	8 bit
Mantissa	7 bit
Precision (relative)	7.8E-03
Max value	3E+38
Min value (normal)	1.7E-38



Can we go lower?

Float8	
Sign	1 bit
Exponent	4 bit
Mantissa	3 bit
Precision (relative)	0.125
Max value	240
Min value (normal)	0.015625

S	Exponent (4 bits)				Fraction (3 bits)		
0	0	1	1	1	0	1	0
7	6	(bit index)		3	2		0

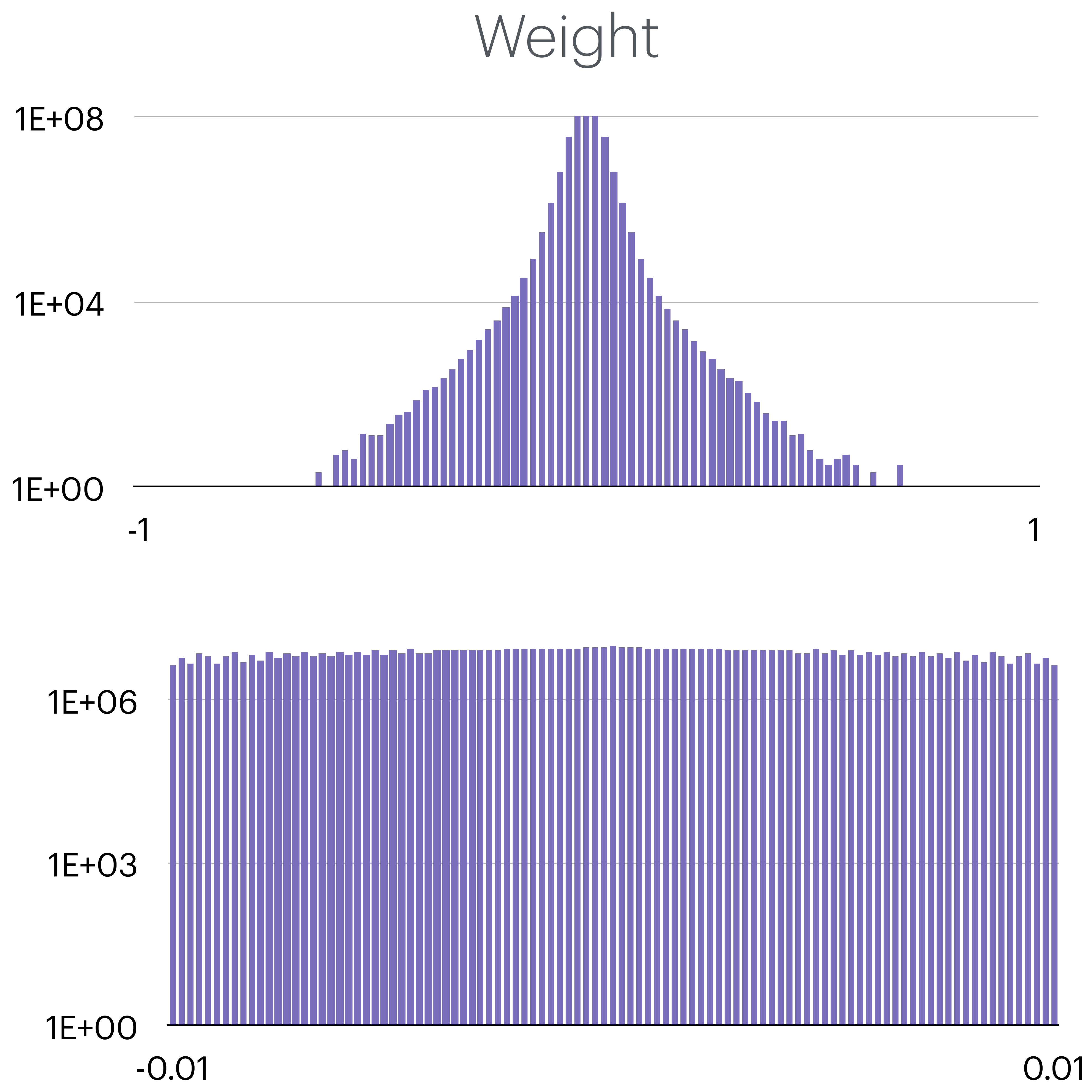
	... 000	... 001	... 010	... 011	... 100	... 101	... 110	... 111
0 0000 ...	0	0.001953125	0.00390625	0.005859375	0.0078125	0.009765625	0.01171875	0.013671875
0 0001 ...	0.015625	0.017578125	0.01953125	0.021484375	0.0234375	0.025390625	0.02734375	0.029296875
0 0010 ...	0.03125	0.03515625	0.0390625	0.04296875	0.046875	0.05078125	0.0546875	0.05859375
0 0011 ...	0.0625	0.0703125	0.078125	0.0859375	0.09375	0.1015625	0.109375	0.1171875
0 0100 ...	0.125	0.140625	0.15625	0.171875	0.1875	0.203125	0.21875	0.234375
0 0101 ...	0.25	0.28125	0.3125	0.34375	0.375	0.40625	0.4375	0.46875
0 0110 ...	0.5	0.5625	0.625	0.6875	0.75	0.8125	0.875	0.9375
0 0111 ...	1	1.125	1.25	1.375	1.5	1.625	1.75	1.875
0 1000 ...	2	2.25	2.5	2.75	3	3.25	3.5	3.75
0 1001 ...	4	4.5	5	5.5	6	6.5	7	7.5
0 1010 ...	8	9	10	11	12	13	14	15
0 1011 ...	16	18	20	22	24	26	28	30
0 1100 ...	32	36	40	44	48	52	56	60
0 1101 ...	64	72	80	88	96	104	112	120
0 1110 ...	128	144	160	176	192	208	224	240
0 1111 ...	Inf	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Can we go lower?

Float8	
Sign	1 bit
Exponent	4 bit
Mantissa	3 bit
Precision (relative)	0.125
Max value	240
Min value (normal)	0.015625

Llama 3.1 8b	
Largest weight	3.21
Smallest weight	-0.83
Largest gradient	5.09
Smallest gradient	-4.41
Largest activation	227.42
Smallest activation	-206.84

S	Exponent (4 bits)				Fraction (3 bits)		
0	0	1	1	1	0	1	0
7	6	(bit index)		3	2		0



Can we go lower?

S	Exponent (2 bits)		Frac. (1 bit)
0	0	1	0
3	2	1	0

Float4	
Sign	1 bit
Exponent	4 bit
Mantissa	3 bit
Precision (relative)	0.125
Max value	240
Min value (normal)	0.015625

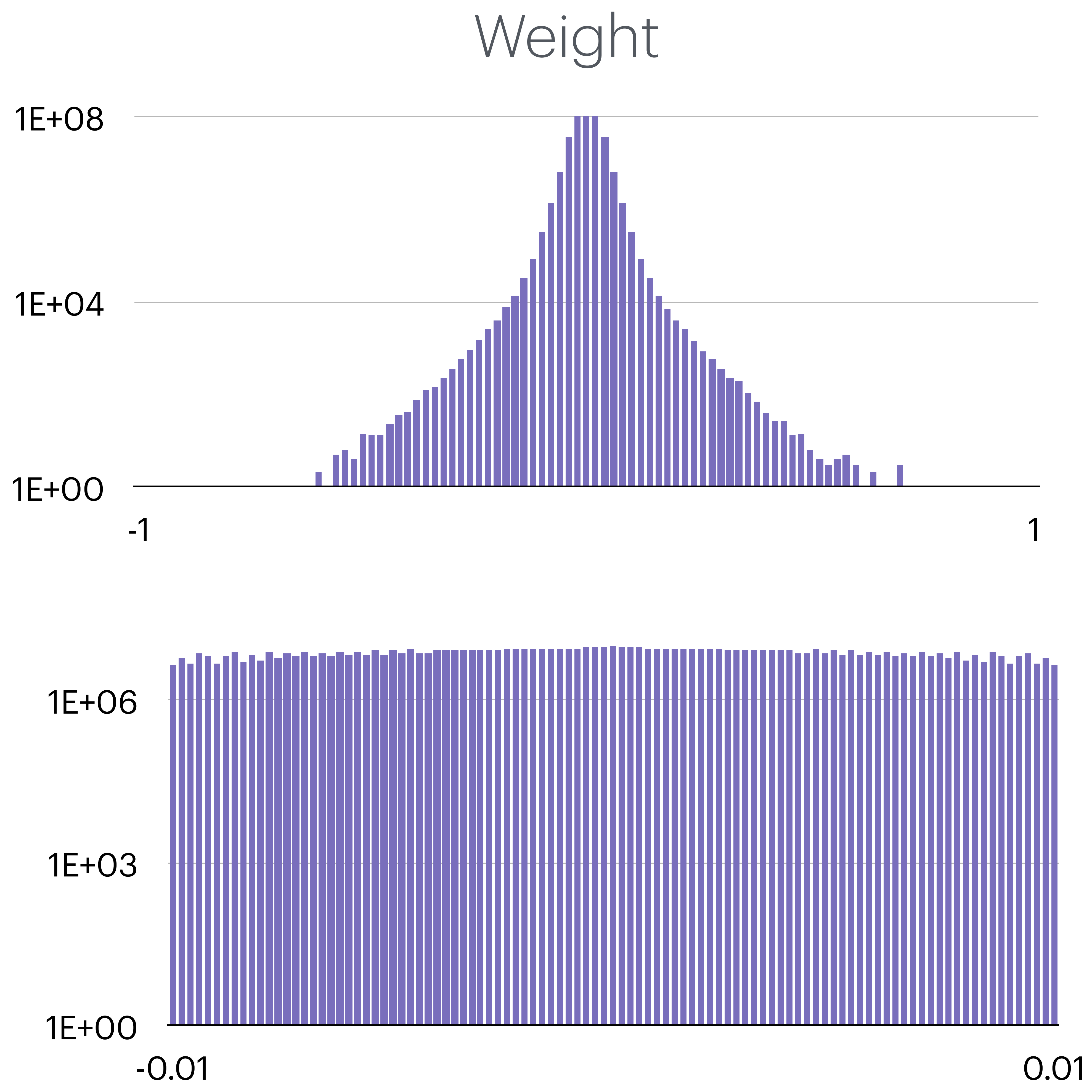
	0 … 0	**0 … 1**	**1 … 0**	**1 … 1**
… 00 …	0	0.5	0	-0.5
… 01 …	1	1.5	-1	-1.5
… 10 …	2	3	-2	-3
… 11 …	Inf	NaN	–Inf	NaN

Can we go lower?

Float4	
Sign	1 bit
Exponent	4 bit
Mantissa	3 bit
Precision (relative)	0.125
Max value	240
Min value (normal)	0.015625

Llama 3.1 8b	
Largest weight	3.21
Smallest weight	-0.83
Largest gradient	5.09
Smallest gradient	-4.41
Largest activation	227.42
Smallest activation	-206.84

S	Exponent (2 bits)		Frac. (1 bit)
0	0	1	0
3	2	1	0



Integer scale quantization

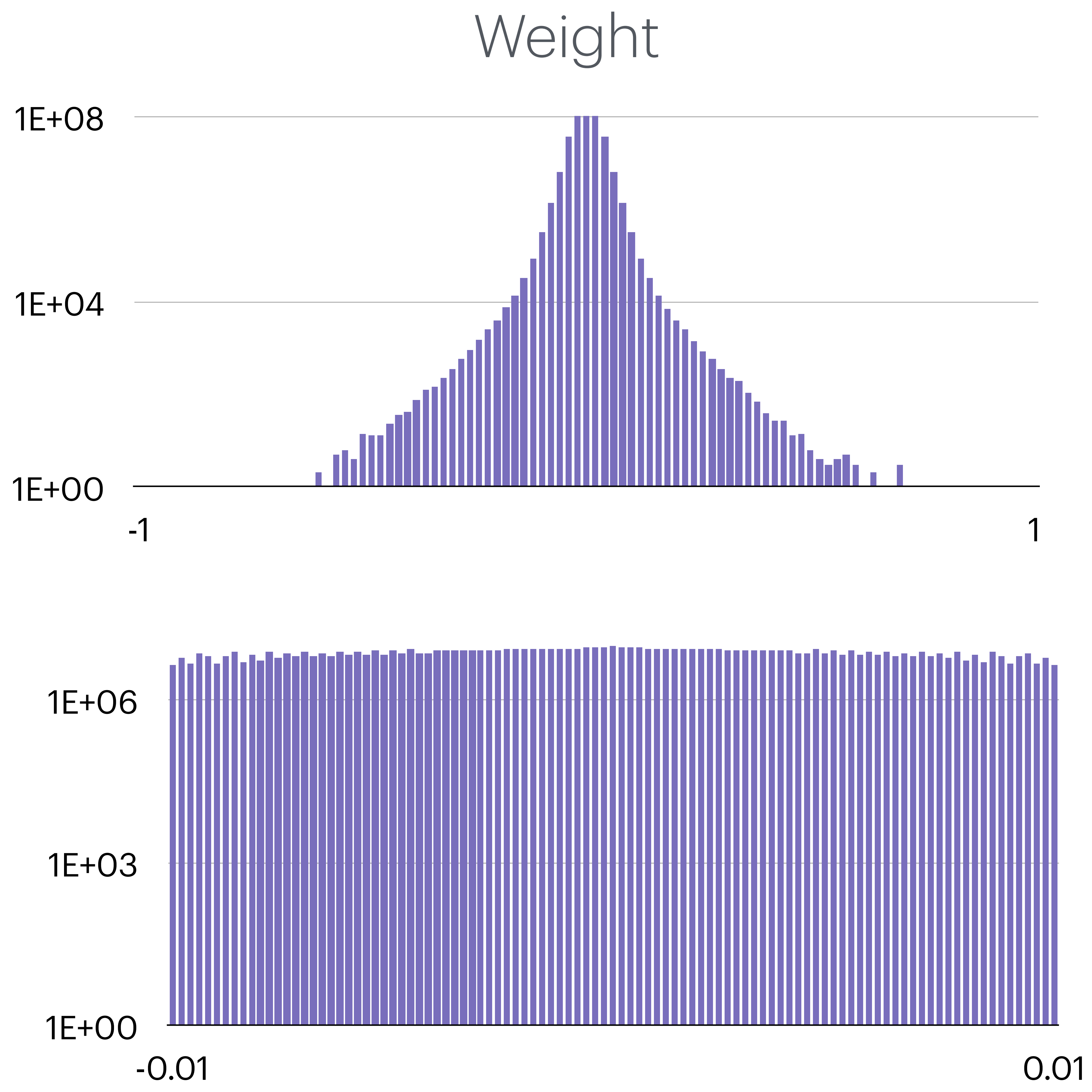
- Find $T = \max_i |W_i|$
- For K-bit integer
  - $Q_i = \underset{q}{\operatorname{argmin}} \left| \frac{W_i}{T}(2^{K-1} - 1) - q \right|$

    for any K-bit signed integer q
  - $Q_i = round\left(\frac{W_i}{T}(2^{K-1} - 1)\right)$

Integer scale quantization

- Quantization error
  - $\dfrac{T}{2^{K-1} - 1}$
- Weights: $\dfrac{NK}{8} + 2$ bytes

Llama 3.1 8b	
Largest weight	3.21
Smallest weight	-0.83
Largest gradient	5.09
Smallest gradient	-4.41
Largest activation	227.42
Smallest activation	-206.84



Integer affine quantization

- Find $A = \min_i |W_i|, \quad B = \max_i |W_i|$

- For K-bit integer

- $Q_i = \operatorname{argmin}_q \left| \frac{W_i - A}{B - A}(2^K - 1) - q \right|$

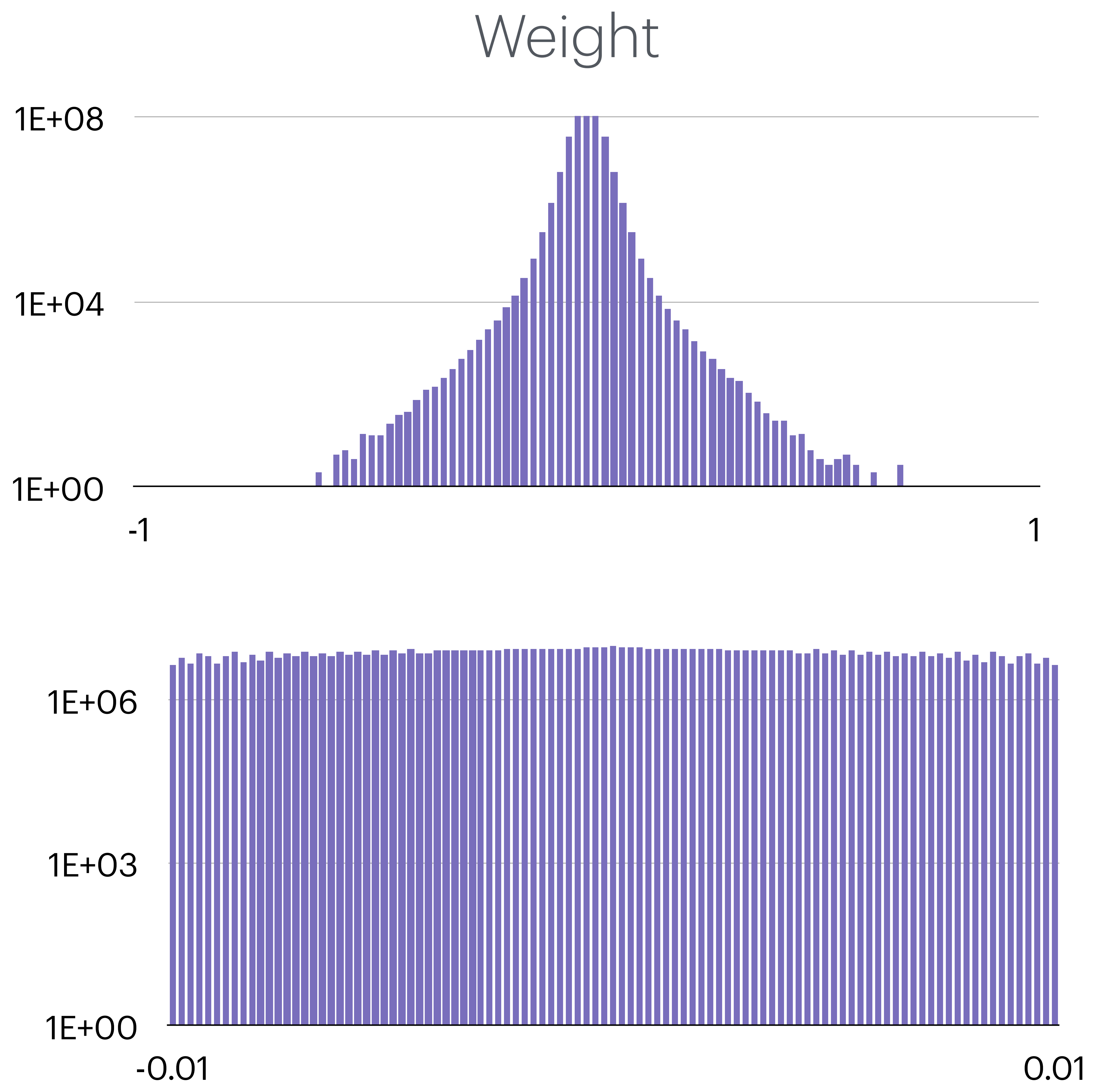
  for any K-bit unsigned integer q

- $Q_i = round\left(\frac{W_i - A}{B - A}(2^K - 1) \right)$

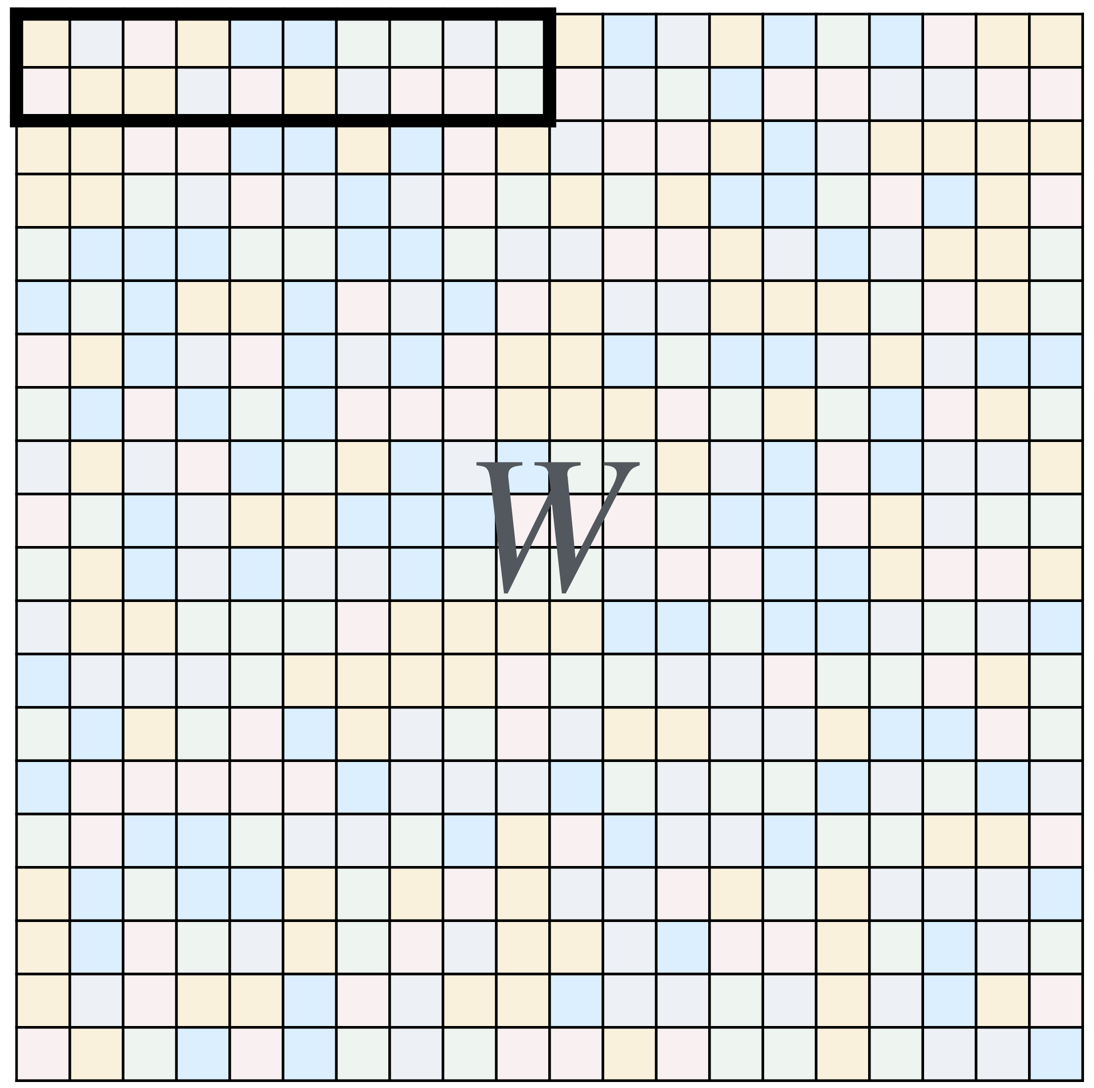
Integer affine quantization

- Quantization error
  - $\dfrac{B-A}{2^K-1}$
- Weights: $\dfrac{NK}{8} + 4$ bytes

Llama 3.1 8b	
Largest weight	3.21
Smallest weight	-0.83
Largest gradient	5.09
Smallest gradient	-4.41
Largest activation	227.42
Smallest activation	-206.84



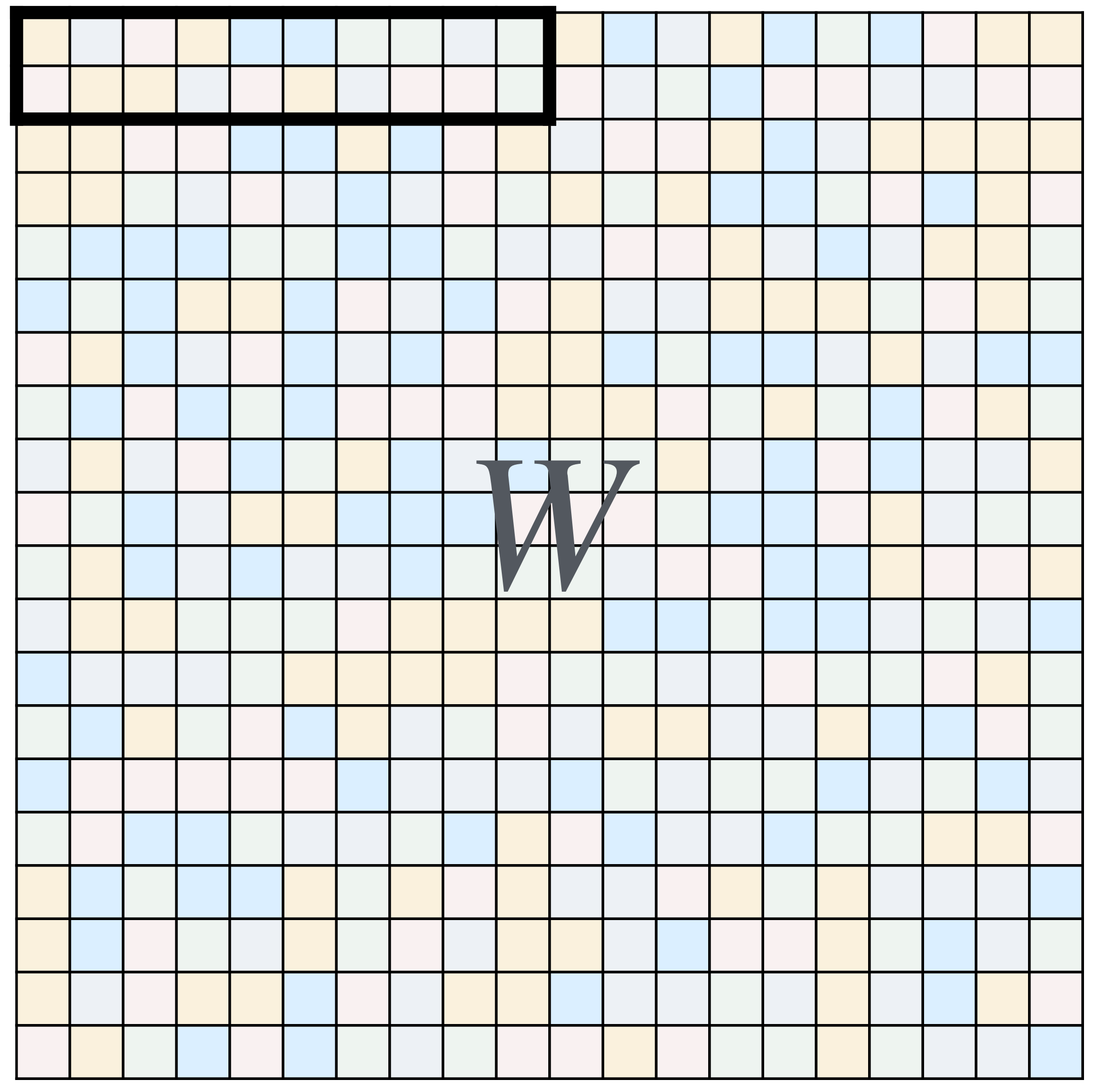
Blockwise Quantization



- Separate quantization constant T, A, B for a block of S weights
- Scale: $\frac{NK}{8} + 2\frac{N}{S}$ bytes
    - $\frac{16}{S}$ extra bits per parameters
- Affine: $\frac{NK}{8} + 4\frac{N}{S}$ bytes
    - $\frac{32}{S}$ extra bits per parameters

[1] Tim Dettmers, et al. 8-bit optimizers via block-wise quantization. 2022

Double Quantization



- Let's quantize the quantization factors T, A, B
- Scale: $\dfrac{NK}{8} + \dfrac{NK}{8S} + 2$ bytes
  - $\dfrac{K}{S}$ extra bits per parameters
- Affine: $\dfrac{NK}{8} + \dfrac{2NK}{8S} + 4$ bytes
  - $\dfrac{2K}{S}$ extra bits per parameters

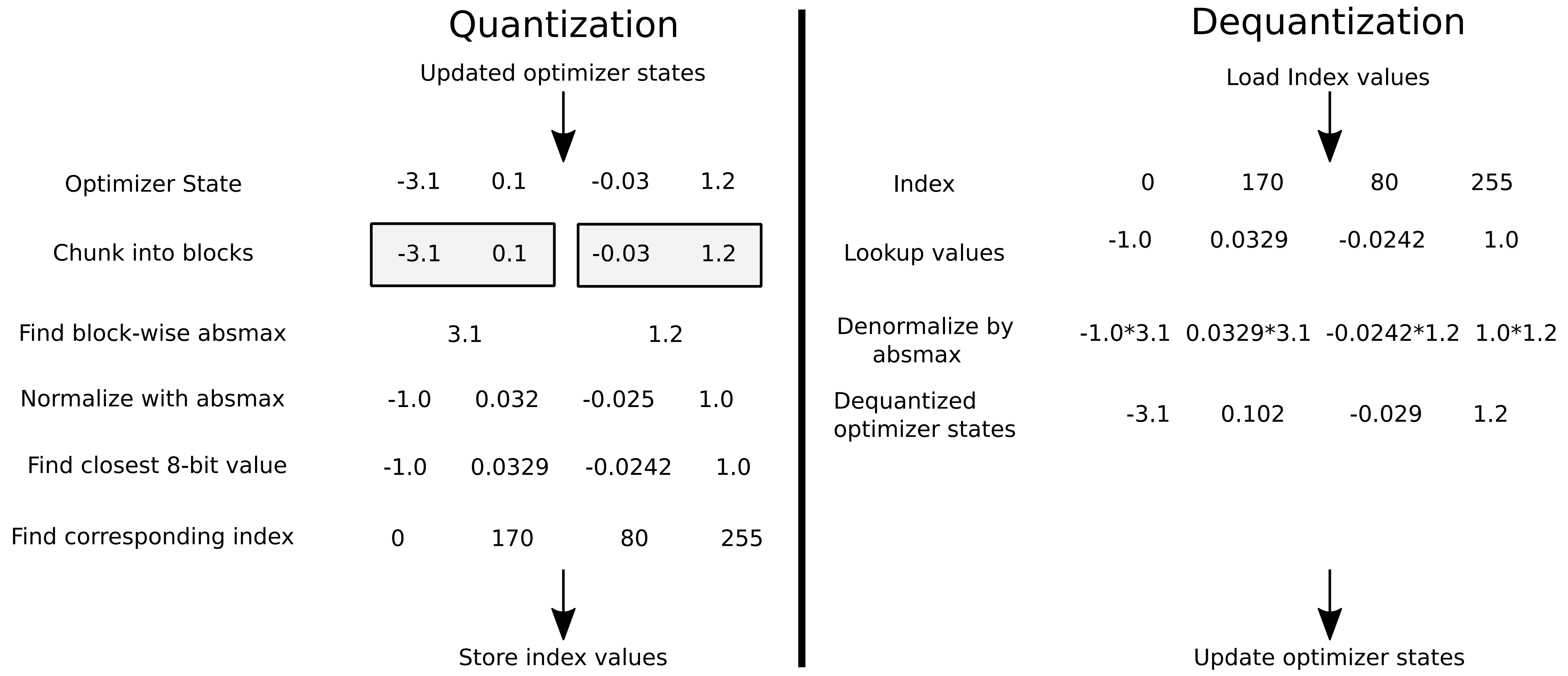
[1] Tim Dettmers, et al. QLoRA: Efficient Finetuning of Quantized LLMs. 2023

Beyond Linear Quantization

- 8-Bit Approximations for Parallelism in Deep Learning : https://arxiv.org/abs/1511.04561
- SqueezeLLM: https://arxiv.org/abs/2306.07629
- Extreme Compression of Large Language Models via Additive Quantization: https://arxiv.org/abs/2401.06118
- PV-Tuning: https://arxiv.org/abs/2405.14852

8-bit Adam

- Quantize 1st and 2nd momentum in Adam
  - 1st momentum: int8
  - 2nd momentum: uint8
  - Non-linear quantization
- Requires "stable" embeddings for LLMs
  - 32-bit optimizer states, normalizations



[1] Tim Dettmers, et al. 8-bit optimizers via block-wise quantization. 2022

Stochastic rounding

How to train with quantized weights?

- Deterministic rounding

$$Q_i = round\left(\frac{W_i}{T}(2^{K-1} - 1)\right)$$

  - Works only if update is largest than quantization

- Stochastic rounding

$$Q_i = sround\left(\frac{W_i}{T}(2^{K-1} - 1)\right)$$

$$sround(x) = \begin{cases} \lfloor x \rfloor + 1 & \text{with } p \leq x - \lfloor x \rfloor \\ \lfloor x \rfloor & \text{otherwise} \end{cases}$$

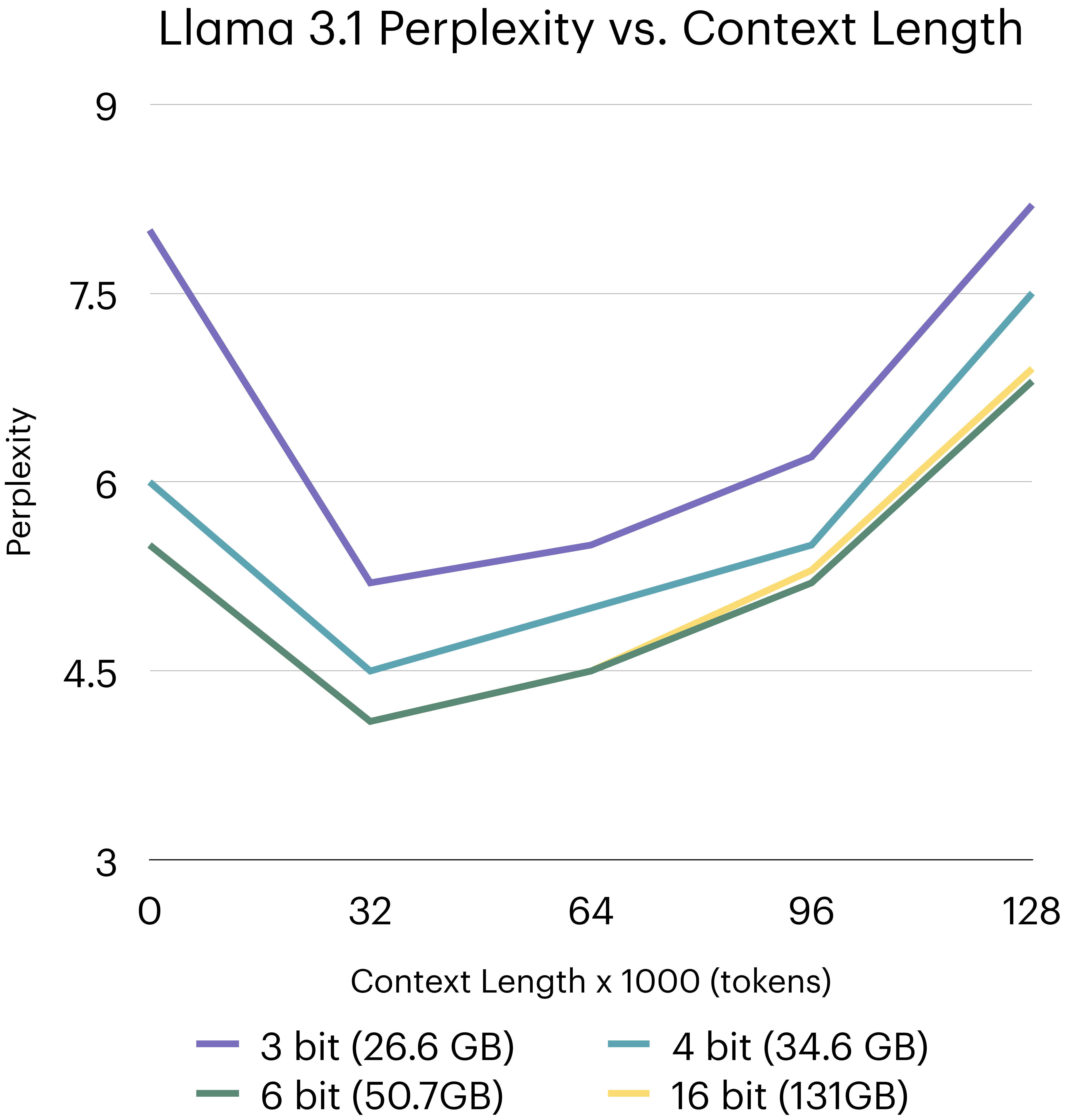
$$E[round(x)] = round(x)$$

$$E[sround(x)] = x$$

[1] Hao Li, et al. Training Quantized Nets: A Deeper Understanding. 2017

How low can we go?

- GPT-style LLM can store about 2 bits of information per parameter
  - Under ideal conditions
- 4 bits in practice
  - Only after training!

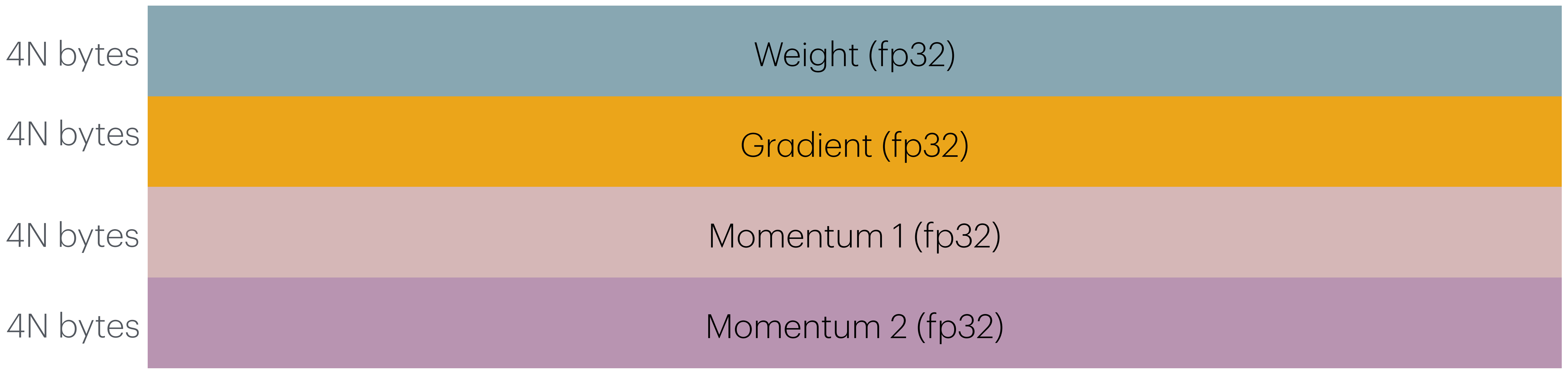


[1] Zeyuan Allen-Zhu, et al. Physics of Language Models: Part 3.3, Knowledge Capacity Scaling Laws. 2024

Training large models

Memory requirements

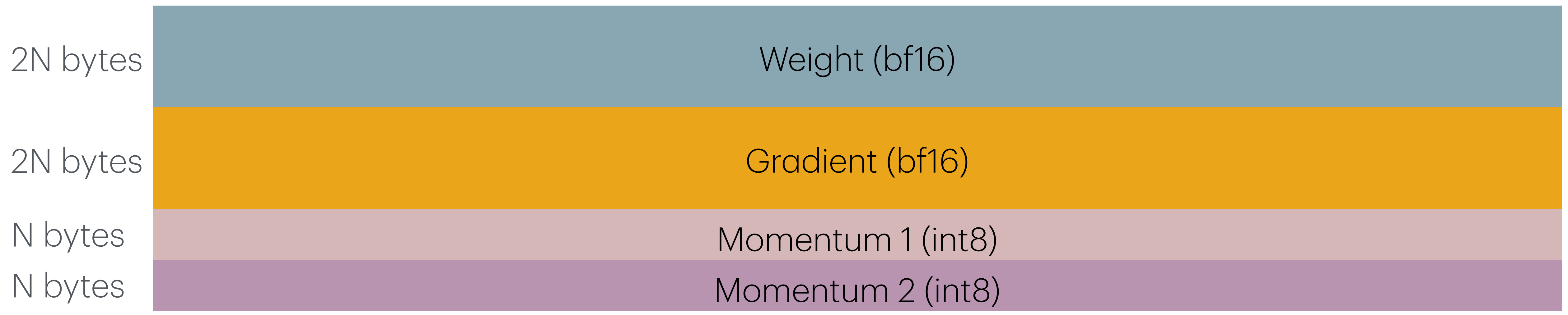
- Without optimization:
  - Model parameters: N
  - Weights: N floats
  - Gradients: N floats
  - Momentum: N floats
  - 2nd momentum (ADAM): N floats
- 16N bytes without counting activations



Training large models

Memory requirements

- Quantization:
  - Model parameters: N
  - Weights: N floats/bfloat16
  - Gradients: N floats/bfloat16
  - Momentum: N uint8
  - 2nd momentum (ADAM): N uint8
- 6N bytes without counting activations



Inference in large models

Memory requirements

- Quantization:
  - Model parameters: N
  - Weights: N int4/int8
- $\frac{1}{2}$N bytes without counting activations

References

- [1] Tim Dettmers, et al. 8-bit optimizers via block-wise quantization. 2022. (link)
- [2] Tim Dettmers, et al. QLoRA: Efficient Finetuning of Quantized LLMs. 2023. (link)
- [3] Hao Li, et al. Training Quantized Nets: A Deeper Understanding. 2017. (link)
- [4] Zeyuan Allen-Zhu, et al. Physics of Language Models: Part 3.3, Knowledge Capacity Scaling Laws. 2024. (link)
- [5] Tim Dettmers. 8-bit approximations for parallelism in deep learning. 2015 (link)
- [6] Sehoon Kim, et al. SqueezeLLM: Dense-and-Sparse Quantization. 2024 (link)
- [7] Vage Egiazarian, et al. Extreme Compression of Large Language Models via Additive Quantization. 2024 (link)
- [8] Vladimir Malinovskii, et al. PV-Tuning: Beyond Straight-Through Estimation for Extreme LLM Compression. 2024 (link)